

Vehicle Modeling and Simulation using Adams/Car

ADM740 Workbook

September 2017

Legal Information

MSC Software Corporation reserves the right to make changes in specifications and other information contained in this document without prior notice. The concepts, methods, and examples presented in this text are for illustrative and educational purposes only and are not intended to be exhaustive or to apply to any particular engineering problem or design. MSC Software Corporation assumes no liability or responsibility to any person or company for direct or indirect damages resulting from the use of any information contained herein.

Copyright © 2017 MSC Software Corporation. All Rights Reserved. This notice shall be marked on any reproduction of this documentation, in whole or in part. Any reproduction or distribution of this document, in whole or in part, without the prior written consent of MSC Software Corporation is prohibited.

The MSC Software corporate logo, Adams, Dytran, Easy5, Fatigue, Laminate Modeler, Marc, Mentat, Patran, MSC, MSC Nastran, Mvision, Patran, SimDesigner, SimEnterprise, SimManager, SimXpert and Sofy are trademarks or registered trademarks of the MSC Software Corporation in the United States and/or other countries. NASTRAN is a registered trademark of NASA. All other trademarks belong to their respective owners.

Contents

Workshop

	<u>Page</u>
1: Open and Run an Assembly.....	WS1-1
2: Templates Vs. Subsystems	WS2-1
3: Creating and Adjusting Subsystems.....	WS3-1
4: Running Suspension Analyses	WS4-1
5: Importing CAD Geometry	WS5-1
6: Running Full-Vehicle Analyses	WS6-1
7: Creating Event Files	WS7-1
8: Tire Testrig Tutorial	WS8-1
9: Road Builder Tutorial	WS9-1
10: Flex Tutorial	WS10-1
11: General Actuation Analysis	WS11-1
12: Template Builder Tutorial	WS12-1
13: Getting Information on Communicators	WS13-1
14: Requests	WS14-1
15: Exploring and Completing Templates	WS15-1
16: Linear Mode Shapes	WS16-1
17: Tilt Table Analysis.....	WS17-1
18: Using Adams/Insight with Adams/Car	WS18-1
19: SVC & SPMM Events in Adams/Car.....	WS19-1

Contents

<u>Appendix</u>	<u>Page</u>
A – Typical Workflow with Adams/Car.....	A-1
B – Adams/Car File.....	B-1
C – Structure of Event File and Driver Control Data	C-1
D – Working with Road Builder.....	D-1
E – Using the Tire Data Fitting Tool (TDFT) in Adams/Car.....	E-1
F – Full-vehicle Assembly.....	F-1
G – Adams/Car Mechatronics.....	G-1
H – Four-post Vertical Excitation Test	H-1
I – Answers.....	I-1

WORKSHOP 1

OPEN AND RUN AN ASSEMBLY

Open and Run an Assembly

- **Problem statement**

- This workshop introduces you to a couple of typical Adams/Car simulations. Adams/Car basically runs either suspension or full-vehicle analyses. Here, you will perform an ISO lane change for a full vehicle. You will also look at the various database related actions in the Adams/Car interface.



This workshop takes about one half hour to complete.

Open and Run an Assembly

- **Setting Up Your Session**

- To create a working directory:

- Depending on the platform you're on, do one of the following:

- On Windows:

- » On your hard drive, create a new folder **acar**. For example, C:\acar.

- On UNIX:

- » To start Adams/Car in your home directory, open an UNIX shell and type **cd**.

- » To create a directory named **acar**, type **mkdir acar**.

- » To move to the new directory, type **cd acar**.

- To start Adams/Car:

- Depending on the platform you're on, do one of the following:

- On Windows, from the Start button, point to **All Programs**, point to **MSC.Software**, point to **Adams 2017.2** and then select **Adams Car**.

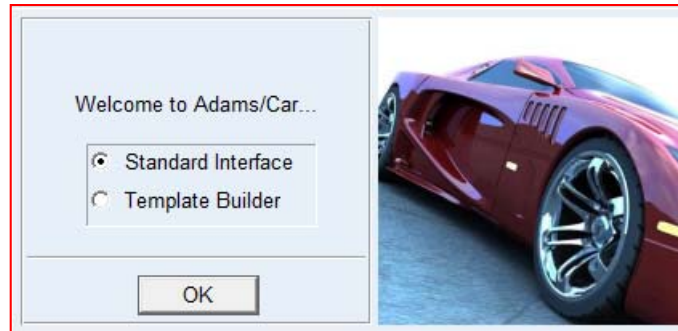
- At your UNIX shell prompt, type **adams2017.2**.

- » From the toolbar, select the **Adams/Car** tool.

- The Welcome dialog box appears 

Open and Run an Assembly

- To toggle to Standard Interface:



- From the Welcome dialog box, select **Standard Interface**, and then select **OK**. (Sometimes the Welcome dialog box contains the option to select a mode, and other times it does not, this depends on the configuration file.) To run analyses, you must be in the Standard Interface mode.
- Once in an Adams/Car session, you can toggle between modes by selecting **Adams/Car Standard Interface** from the **Tools** menu. If Adams/Car Template Builder is not listed, then you are already in the Standard Interface mode.

Open and Run an Assembly

- **To set the working directory:**

1. From the **File** menu, select **Select Directory**.
2. Select the **acar** directory you just created.
3. Select **OK**.

Adams/Car sets the working directory where it will write your output files.

- **To create a new database and set it as the writable database:**

1. From the **Tools** menu, point to **Database Management**, and then select **Create Database**.

The Create New Database dialog box appears.

2. In the **Database Alias** text box, enter **acar_training**.
3. In the **Database Path** text box, explicitly define the path for the new database. For Example:

- On Windows: c:\acar\acar_training.cdb
- On UNIX: /acar/acar_training.cdb

Adams/Car creates a new database named acar_training.cdb.

Open and Run an Assembly

4. Select **OK**.

This creates the directory and adds it to your list of searchable databases in your Adams/Car session.

5. To verify that this database has been added to your **session**, from the **Tools** menu, point to **Database Management**, and then select **Database Info**.
6. From the **Tools** menu, point to **Database Management**, and then select **Set Default Writable**.
7. Make sure that **Database Name** is set to **acar_training** (select the down arrow and then select **acar_training**).
8. Select **OK**.

Adams/Car automatically saves the configuration file.

Open and Run an Assembly

- **Open .acar.cfg**
 - Locate .acar.cfg file on your machine
 - By default it would be in the directory pointed to by the HOME system environment variable. Look through it and note the new **acar_training** database and its path listed.
 - Note following entry:
 DATABASE acar_training C:/acar/acar_training.cdb

Open and Run an Assembly

- In the above slides we created a Database and then added it to our Car session.
- The course instructor will provide you with a ready database **acar_training_MD.cdb**, which you will need to add to your car session. You will need the files in this database later as you proceed through the course in some of the other sections.
 - Follow the steps below to add the database to the session:
 1. Unzip the provided **acar_training_MD.cdb.zip** to your Adams car folder.
 2. From the **Tools** menu, point to **Database Management**, and then select **Add to Session**. The **Add Database to session** dialog box appears.
 3. In the **Database Alias** text box, enter **acar_training_MD**.
 4. In the **Database Path** text box, browse to the location where you keep this database; preferably in the acar folder itself.

Open and Run an Assembly

- **Simulating a full-vehicle assembly**
 - You first open a full-vehicle assembly and then perform a full-vehicle, ISO lane-change analysis with the Driving Machine. You then investigate the results by animating the assembly. The animation is based on the results of your analysis.
- **To open a full-vehicle assembly:**
 1. From the **File** menu, point to **Open**, and then select **Assembly**.
 2. Right-click the **Assembly Name** text box, point to **Search**, select **<acar_shared>\assemblies.tbl**, and then select **MDI_Demo_Vehicle.asy** then Click **Open**.
 3. Select **OK**.

In the Message window, Adams/Car informs you when the assembly is ready. The messages in this window are important and will be discussed in detail in subsequent chapters.
 4. Close the Message window.

Open and Run an Assembly

- **To perform an analysis:**


1. From the **Simulate** menu, point to **Full-Vehicle Analysis**, point to **Course Events**, and then select **Double Lane Change**.
2. In the **Output Prefix** text box, enter **workshop1a**.
3. In the **Output Step Size** text box, enter **0.1**.
4. In the **Vehicle Velocity** text box, enter **70**. The default units should be km/hr.
5. Select **OK**.

In the Message window, Adams/Car informs you about the progress of the analysis and when the simulation is complete.

5. Close the Message window.

Open and Run an Assembly

- **To investigate the results:**


1. From the **Review** menu, select **Animation Controls**.
2. To animate the assembly, select the **Play** tool  .
3. Zoom out to see more of the road grid:
 - Type a lowercase **z**.
 - Hold down the left mouse button, and do either of the following:
 - » To enlarge the display of the assembly, or zoom in, move the cursor up.
 - » To shrink the display of the assembly, or zoom out, move the cursor down.
 - To exit zoom mode, release the mouse button.

- **To trace the motion of the vehicle:**

1. In the Animation Controls dialog box, change **No Trace** to **Trace Marker**.
2. Right-click the **Trace Marker** text box, point to **Marker**, and then select **Browse**.
The Database Navigator appears.
3. Double-click **MDI_Demo_Vehicle**, double-click **TR_Body**, double-click **ges_chassis**, and then select **cm**.
4. Select **OK**.

Open and Run an Assembly

- **To follow the car in the animation:**

1. Change **Fixed Base** to **Base Part**.
2. Right-click the **Base Part** text box, point to **Body**, and then select **Pick**.
3. Move the cursor over the vehicle and right-click to see a list of parts in the area of the cursor.
4. Select any of the vehicle's parts; for example, you can select ges_chassis.
5. Select the **Play** tool .



The camera should move with the car, as the white line traces the path of the body marker.

6. Close the Animation Controls.

Open and Run an Assembly

- **To animate the full-vehicle from the driver's perspective:**
 1. From the **Review** menu, select **Postprocessing window** or press **F8**.
 2. Right-click anywhere on the main window of the PostProcessor screen, and then select **Load Animation**.
Database Navigator opens.
 3. Select **workshop1a_dlc**, and then select **OK**.
Adams/PostProcessor loads the animation into the viewport.
 4. Press **Ctrl + f** to fit the vehicle to the window.
 5. Press the **c** key and pick a point on the vehicle as the new center of rotation.
 6. To rotate the vehicle to look at the rear, press the **r** key and hold down the left-mouse button while moving the mouse to rotate the vehicle.

Open and Run an Assembly

7. Right-click the **Dynamic Translate (xy)** tool .
8. Select the **Dynamic Translate (z)** tool .
9. Hold down the left mouse button while moving the mouse to translate the vehicle.
10. For more view manipulation options, right-click an empty area in the viewport. Each option is listed with its respective hotkey.
11. Select the **Camera** tab. Right-click the **Follow-Object** text box, point to **Part**, and then select **Pick**.
12. Select any part on the vehicle just as you did in the previous section, in Step 3 on slide 11 (for example, the steering wheel).
Note: If you do select steering wheel, skip Step 13.
13. To lock the view to the part you selected in Step 12, select **Lock Rotations**.
14. To animate from this new perspective, select the **Play** tool.
15. If necessary, adjust the speed of the animation using the **Speed Control** slider in the **Animation** tab.
16. Press **F8** to return to Adams/Car.

Adams/Car Database and Working Directory

- Investigate the output files created with the run:
- **Save Assembly**
 1. From the **File** Menu , point to **Save As – Assembly**
 2. The New Assembly Name : **MDI_Demo_Vehicle_saved**
 3. Make sure Target Database: **acar_training**
 4. Check “**Close Assembly after Save**”. Click **OK**.
 5. Browse to the **C:\acar\acar_training.cdb\assemblies.tbl** using Windows Explorer
 6. Open the **MDI_Demo_Vehicle_saved.asy** file with a text editor and keep it open.
 7. Now go to **C:\acar\acar_training.cdb\subsystems.tbl**.

Adams/Car Database and Working Directory

- **To Copy one assembly to another database so that your colleague gets a fresh copy to work with:**
 1. From the **File** menu, point to **Open**, and then select **Assembly**.
 2. Right-click the **Assembly Name** text box, point to **Search**, select **<acar_shared>\assemblies.tbl**, and then select **MDI_Demo_Vehicle.asy**.
 3. Select **OK**.
 4. From the **Tools** menu, point to **Database Management**, and then select **Publish Assembly**
 5. Fill the box in as follows:
 - **File Name: MDI_Demo_Vehicle**
 - **Publish To: Database**
 - **Write Mode: Overwrite**
 - **Target Database: acar_training**
 - **Check Include Template file**
 - **Check Update Session Data with 'Target Database'**
 - **Select OK**

Adams/Car Database and Working Directory

6. Browse to **C:\acar\acar_training.cdb\assemblies.tbl**.
7. Open the **MDI_DEMO_Vehicle.asy** file with a text editor. Compare this with the previous **MDI_Demo_Vehicle_saved.asy**.

*Look at how the different subsystems are being referenced in the two assembly *.asy files. One references the shared database for the subsystem files and the other references the subsystems in acar_training database.*

Optional - Open and Run a Suspension Assembly

- **Simulating a suspension assembly**
 - You simulate a suspension assembly in the same way you simulated the full-vehicle assembly.
- **To open a suspension assembly:**
 1. From the **File** menu, point to **Open**, and then select **Assembly**.
 2. Right-click the **Assembly Name** text box, point to **Search**, select **<acar_shared>\assemblies.tbl**, and then select **mdi_front_vehicle.asy**.
 3. Select **OK**.

In the Message window, Adams/Car informs you when the vehicle assembly is ready.
 4. Close the Message window.

Open and Run a Suspension Assembly

- **To perform a parallel wheel travel suspension analysis:**
 1. From the **Simulate** menu, point to **Suspension Analyses**, and then select **Parallel Wheel Travel**.
 2. Set up the analysis:
 - **Output Prefix:** workshop1b
 - **Number of Steps:** 10
 - **Bump Travel:** 100
 - **Rebound Travel:** -100

The default units for Bump and Rebound Travel should be mm (Settings → Units).
 3. Select **OK**.
 4. When the analysis is complete, close the **Message** window.

Open and Run a Suspension Assembly

- **To review the results by animating your assembly:**
 1. From the **Review** menu, select **Animation Controls**.
 2. Change **Base Part** to **Fixed Base**.
 3. Change **Trace Marker** to **No Trace**.
 4. Select **Play**.
- **To trace the path of a vehicle's marker:**
 1. Change **No Trace** to **Trace Marker**.
 2. Right-click the **Trace Marker** text box, point to **Marker**, and then select **Browse**.

The Database Navigator appears.
 3. Double-click **MDI_FRONT_SUSPENSION**, double-click **gel_spindle**, and then double-click **cm**.
 4. Select **Play**.
 5. Zoom in to look for the white line that traces the path of the **gel_spindle.cm** marker.

WORKSHOP 2

TEMPLATES VERSUS SUBSYSTEMS

Templates Versus Subsystems

- **Problem statement**

- Understanding the **difference between a template and a subsystem** in Adams/Car is a pivotal first step toward using the full power of Adams/Car. To illustrate this, consider two people working side by side, both on a steering system. Looking at both computer screens, you see what appears to be the same model. However, one user is working on a template in template-builder mode, while the other is working on a subsystem in standard-interface mode. So, what's the difference?
- As described before, the difference is what you can do with the models. Each part and its topology, or the way that information and parts are connected, is defined in **Template Builder using parameters (variables)**, while the specific values of those parameters are set in Standard Interface. Additionally, analyses can only be performed in Standard Interface, based on a model (specifically, a template) created in Template Builder. A good way to understand this distinction is to create a template file and a subsystem file and compare their contents.



This workshop takes about one half hour to complete.

Templates Versus Subsystems

- **Opening a template file**
 - In this workshop, you create an ASCII template file and compare it to a subsystem file. Template files can exist either as binary or text (ASCII) files. By default, the templates saved in Adams/Car are binary, so to view the contents, you must save this one as text.
- **To choose the template-builder mode in Adams/Car:**
 - From the **Tools** menu, select **Adams/Car Template Builder**.

Note: You can toggle between Template Builder and Standard Interface by pressing F9.

Templates Versus Subsystems

- If Adams/Car Template Builder is not an option under the Tools menu, you must set your user privileges to expert:

1. Exit Adams/Car.
2. Modify your private **.acar.cfg** file to include: ENVIRONMENT MDI_ACAR_USERMODE expert

Tip: For information on where the .acar.cfg file is saved, see Saving Configuration Files, in Section 1.

```
!-----!  
! - List of personal environment variables  
!-----!  
! Desired user mode (standard/expert)  
ENVIRONMENT MDI_ACAR_USERMODE expert  
!  
!-----!
```

3. Start Adams/Car.
4. Set the working directory to acar, just as you did in *To Set the Working Directory*, in Workshop 1.

Templates Versus Subsystems

- To open the MacPherson suspension template from the shared database:
 1. From the **File** menu, select **Open**.
 2. Right-click the **Template Name** text box, point to **Search**, and then select **<acar_shared>/templates.tbl**.
 3. Double-click **_macpherson.tpl**.
 4. Select **OK**.

Templates Versus Subsystems

- **To change mass of the part in the template builder:**
 1. Right-click on the empty workspace of Adams/Car template builder, and then select **Front**. This would set your model in the front view.
 2. Look for the general part name **gel_lower_control_arm**, Right-click on that part, and then select **modify**. (Or From the Model Browser given on left hand side of window, Point to **Browse**, then select **Parts > General Parts**, Right click on **gel_lower_control_arm**, select **Modify**.)
 3. Set value of **Mass** from default value to say **10** in **Modify General Part** dialog box.
 4. Click **OK**.

Note: The modification of mass value is to demonstrate the unidirectional flow of information from template file to subsystem file and that will be discussed after taking a look at the subsystem file.

Templates Versus Subsystems

- **To save the template file as a text file:**

1. From the **File** menu, select **Save As**.
2. In the **New Template Name** text box, enter **mac_ascii**. (Note that this text box is grayed-out because it is not a required text box to perform this function. If you do not enter a name, Adams/Car saves the file with its current name.)
3. Set **File Format** to **Ascii**. You need to do this to be able to read the file.
4. Select **OK**.

Adams/Car saves the file in the **acar_training** database, which you set up earlier.

Templates Versus Subsystems

- **To open the template file and look at the contents:**

1. Open **_mac_ascii** in a text editor.

Depending on the platform you're working on, **_mac_ascii** should be located in:

- Windows: C:/acar/acar_training.cdb/templates.tbl
 - UNIX: /acar/acar_training.cdb/templates.tbl
2. If you did not set **acar_training** as the default writable database in Workshop 1 - Open and Run an Assembly, use **Tools → Database Management → Database Info** to find out which database is the default writable database.

Templates Versus Subsystems

- The file has all the information needed to define the model using markers, parts, communicators, forces, and so on. Take a look at the file to see what kind of information is stored.

For example, the following is the beginning of the definition of the **left lower control arm** (near the top):

```
!----- gel_lower_control_arm -----!  
!  
!  
defaults coordinate_system &  
  default_coordinate_system = ._mac_ascii.ground  
!  
part create rigid_body name_and_position &  
  part_name = ._mac_ascii.gel_lower_control_arm &  
  location = 0.0, -550.0, 150.0 &  
  orientation = 0.0d, 90.0d, 180.0d
```

- Look at the value of **mass** of part **lower control arm**.
- Notice the commands to create parts, joints, markers, and so on. It is not important that you understand everything in this file, just that you get a sense of what type of information is stored here.

Templates Versus Subsystems

- **Creating a subsystem file**
 - Subsystem files can only exist as text (ASCII) files, so you do not need to convert from binary.
- **To create the subsystem file:**
 1. Select **Adams/Car Standard Interface** from the **Tools** menu or press **F9** to return to the Standard Interface.
 2. From the **File** menu, point to **New**, and select **Subsystem**.
 3. In the **New Subsystem** box, enter **front_suspension_ascii** name in the **Subsystem Name** field.
 4. Select front as minor role.
 5. Right-click the **Template Name** text box, point to **Search**, and then select **<acar_training>**.
 6. Double-click the directory, **template.tbl**, and then double-click **_mac_ascii.tpl**.
 7. Select **OK**. If asked, you may choose to use the template stored in memory.

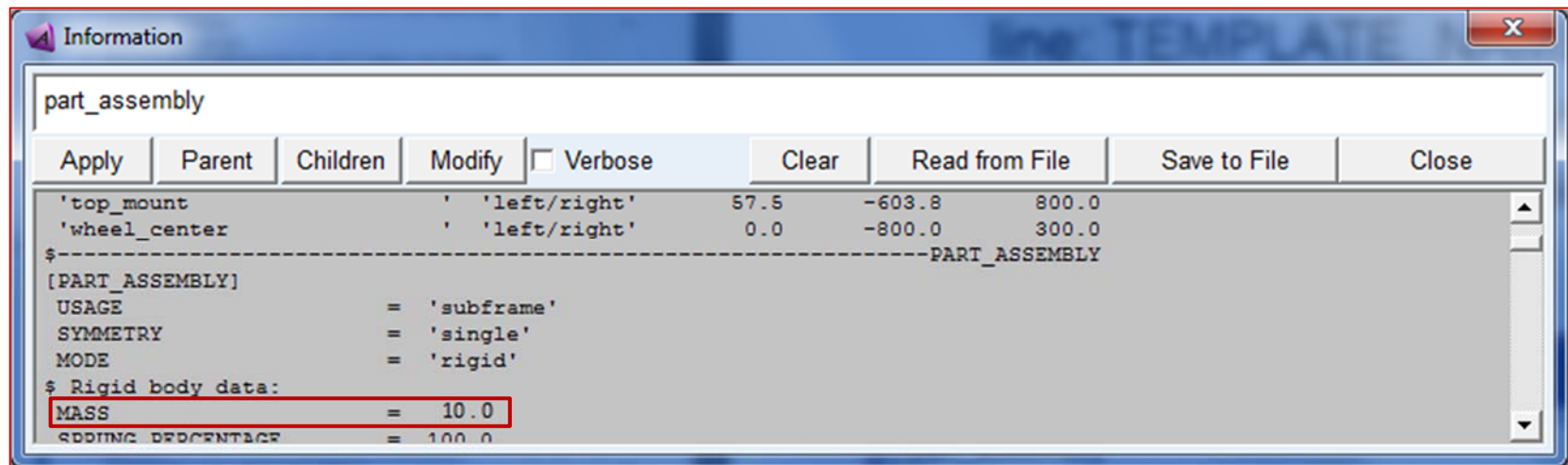
Templates Versus Subsystems

- **To save the subsystem file:**
 - From the **File** menu, select **Save As**.
 - Select **OK**.
- **To look at contents of the subsystem file:**
 1. From the **Tools** menu, select **Show File**.
 2. Right-click the **File Name** text box, point to **Search**, and then select **<acar_training>**.
 3. Double-click the directory, **subsystems.tbl**, and then double-click **front_suspension_ascii.sub**.
 4. Select **OK**.

The Information window displays the contents of **front_suspension_ascii.sub**.

Templates Versus Subsystems

5. Take a look at the file to see what kind of information it stores. The top portion of the file looks very similar to the template file, but the rest is very different, as it resets values of parameters in the template file.
6. Notice that in the [SUBSYSTEM_HEADER] section, the MacPherson information is referenced for loading into your Adams/Car session with the line: `TEMPLATE_NAME = '<acar_shared>/templates.tbl/_macpherson.tpl'`.
7. Also, notice that the subsystem sets the values for the parameters in the lower control arms: look at the **mass** value, it just get inherited from template file.



Templates Versus Subsystems

- **To change mass of the part in the subsystem file:**
 1. Right-click on the empty workspace of Adams/Car standard interface, and then select **Front**. This would set your model in the front view.
 2. Look for the general part name **gel_lower_control_arm**, Right-click on that part, and then select **modify**.
Alternatively, From Model Browser given on left hand side of window, Point to Browse, then select Parts>General Parts, Right click on **gel_lower_control_arm**, select **Modify**.
 3. Set value of **mass** from default value to say **20** in **Modify General Part** dialog box.
 4. Click **OK**.
 5. From the **File** menu, select **Save As**.
 6. Select **OK**.

Note: The modified value of mass gets stored in the subsystem file and won't affect the value stored in the template file. You are going see that next.

Templates Versus Subsystems

- **Now, look at contents of the template file:**
 1. Open **_mac_ascii** in a text editor.
Depending on the platform you're working on, **_mac_ascii** should be located in:
 - Windows: C:/acar/acar_training.cdb/templates.tbl
 - UNIX: /acar/acar_training.cdb/templates.tbl
 2. Search for the general part name **gel_lower_control_arm**, and then look at the **mass** value.

Note: Please note that the mass value does not changes in the template file. This demonstrated that flow of the information is unidirectional. That is from Template file (.tpl) → Subsystem file (.sub) → Assembly file (.asy).

Templates Versus Subsystems

- **Summary**

- Overall, a template defines the structure/topology of a model, and a subsystem redefines parameters to create an instance of the template.
- Below is a table that lists the characteristics of the two file types:

Table 1: Comparison of Templates and Subsystems

Characteristic:	Templates:	Subsystems:
Used to define structure of model	Yes	No
References the other file (template versus subsystem)	No	Yes
Can be edited to change topology (for example, the point at which force is applied)	Yes	No
Can edit parameters which define the model	Yes (set default)	Yes (has priority)
Are used to define an assembly directly	No	Yes (in an assembly)

Templates Versus Subsystems

- An Adams/Car template is an Adams model built by an expert Adams/Car user in the Adams/Car Template Builder. The Adams/Car template contains geometric and topological data. The template file can be stored in ASCII or binary format.
- An Adams/Car subsystem is based on an Adams/Car template and allows the standard user to alter the geometric data and some of the topological data of the template. The subsystem file is stored in ASCII format.
- An Adams/Car assembly is a number of Adams/Car subsystems assembled together with an Adams/Car test rig. The assembly file is stored in ASCII format and is a list of the subsystems and test rig associated with the assembly.

WORKSHOP 3

CREATING AND ADJUSTING SUBSYSTEMS

Creating and Adjusting Subsystems

- **Problem statement**

- In this workshop, you create a new subsystem and learn how to adjust its parameters.



This workshop takes about one half hour to complete.

Creating and Adjusting Subsystems

- **Creating and saving a subsystem**
- **To create a new subsystem:**
 1. Change to Adams/Car Standard Interface.
 2. From the **File** menu, point to **New**, and then select **Subsystem**.
 3. In the **Subsystem Name** text box, enter **my_macph**.
 4. Set **Minor Role** to **rear**.
 5. Right-click the **Template Name** text box, point to **Search**, and then select **<acar_shared>\templates.tbl**.
 6. Double-click **_macpherson.tpl**.
 7. Select **OK**.

Adams/Car displays the subsystem.

Creating and Adjusting Subsystems

- **To save the subsystem:**

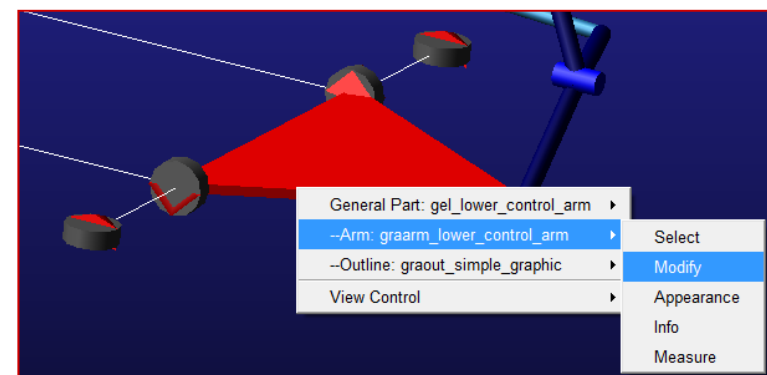
1. From the **File** menu, point to **Save As**, and then select **Subsystem**.

The Save Subsystem dialog box appears. Because only one subsystem is open in Adams/Car, by default my_macph is selected. However, if you have more than one subsystem opened in your session, use the down arrow to select which one you would like to save. Adams/Car saves the subsystem in the database, **acar_training**, which was set as the default database in Setting Up Your Session, in Workshop 1. Adams/Car saves the file in the subsystems.tbl table.

2. Select **OK**.


Creating and Adjusting Subsystems

- Modifying a subsystem
- Geometry and Mass relation:
 1. From the Model Browser, point to **Parts > General Parts**, right click the **General Part:Gel_Lower_control_arm**, select **Info** and note the mass of the lower arm. Alternately, Right Click the General Part: **Gel_Lower_control_arm – Info** and note the mass of the lower arm.
 2. Now look for **--Arm:graarm_lower_control_arm** and **modify** the Arm geometry
 3. Change the **Thickness** to **33.3** and click **OK**
 4. Check mass of the General Part: Gel_Lower_control_arm. Does it change with the change in thickness? Why?



Creating and Adjusting Subsystems

- **To adjust mass properties:**

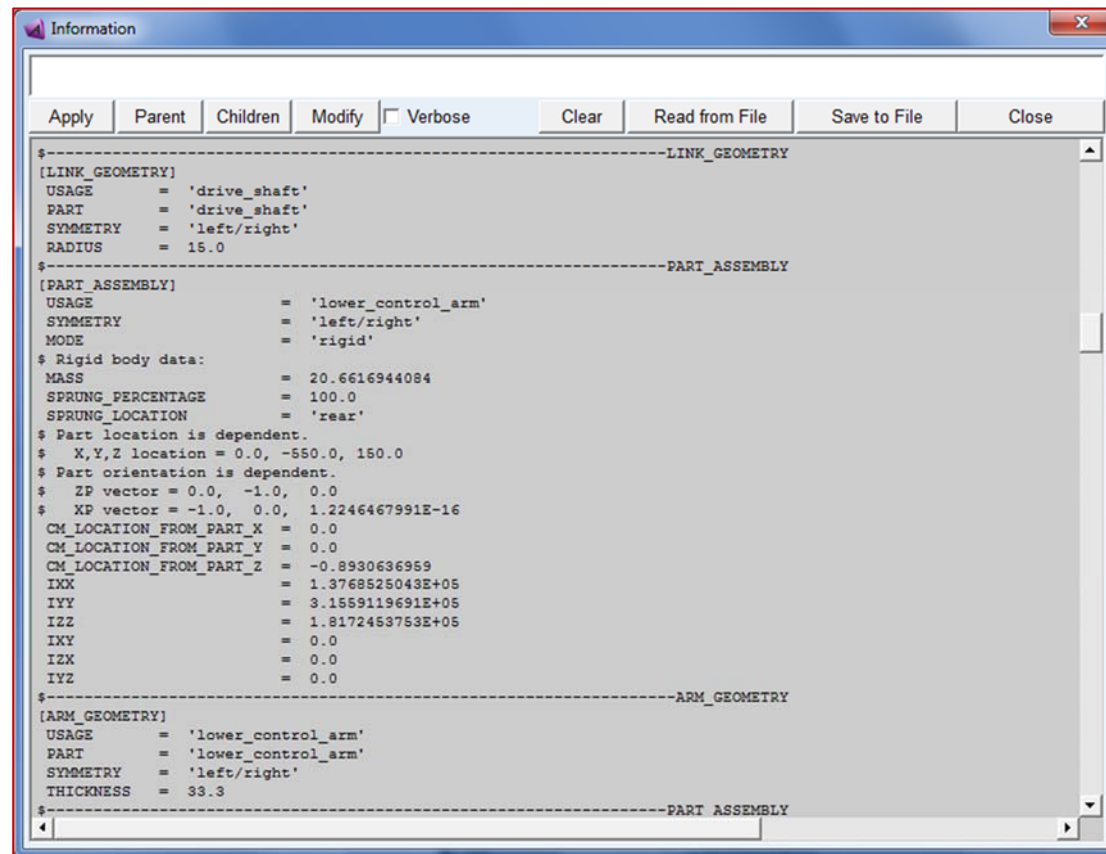
1. From the Model Browser right-click the General Part, **gel_lower_control_arm**, and select **Modify**.
2. To calculate the mass properties of the lower control arm based on the new geometry, select the **Calculate Mass** tool .
3. Note that the **Material** option is selected for **Density** and **Steel** is selected as the **Material Type** drop down menu, therefore the density value for **Steel** is used to calculate mass.
4. To apply the changes to the part, select **OK**.
5. From the **File** menu, select **Save** to save the subsystem again.
6. To save a backup copy, enter **Yes**.

Creating and Adjusting Subsystems

7. From the **Tools** menu, select **Show File**.

8. Right-click and search for
<acar_training>\subsystems.tbl\my_macph.sub.

Adams/Car displays the new mass and inertia values for the lower control arm:



Creating and Adjusting Subsystems

- **To adjust hardpoints:**

1. From the **Adjust** menu, point to **Hardpoint**.

Note that you could select either of the following:

- **Modify** - Lets you adjust hardpoints one at a time
- **Table** - Lets you adjust all hardpoints.

Alternatively, you can modify **Hardpoint** from Model Browser, expand **Hardpoints** box, right-click the hardpoint you want to modify, and then select **Modify**.

2. Select **Table**.
3. Set **Display** to **Both**.

Note: If necessary, resize the dialog box to see all hardpoints.

Creating and Adjusting Subsystems

- **To edit the size of the lower control arm:**
 1. Change **loc_x** for either **hpl_lca_front** or **hpr_lca_front** from **-200** to **-120**. Because these hardpoints are symmetrically parameterized, changing one will automatically change the other.
 2. Select **Apply**.
 3. Change **loc_x** for either **hpl_lca_rear** or **hpr_lca_rear** from **200** to **120**. Because these hardpoints are symmetrically parameterized, changing one will automatically change the other. To display the new value in the table, press **Enter**.

Note: The hardpoint does not change location yet, but it will after you select **Apply** or **OK**.

4. Select **Apply**.
The lower control arm becomes smaller.

Creating and Adjusting Subsystems

- **To close the hardpoint table:**

1. To close the table, select **OK** or **Cancel**. Only when you save your subsystem will Adams/Car update the .sub file with these new hardpoint values.

Note: Because you have not saved your subsystem yet, Adams/Car has not updated your subsystem file (my_macph.sub).

- **To adjust parameter variables:**

1. From the **Adjust** menu, point to **Parameter Variable**, and then select **Modify**. Alternatively, you can select **Parameters** from **Model Browser**, right-click the **Parameter Variable** you want to modify, and then select **Modify**.
2. To see what parameter variables are available in this template, right-click the **Parameter Variable** text box, point to **Variable**, and then select **Guesses**.

Here, you can see that one parameter available for adjustment is, pvl_toe_angle (toe angle). Toe angle is the angle between the longitudinal axis of the vehicle and the line of intersection of the wheel plane and the road surface. Adams/Car reports toe angle in degrees. It is positive if the wheel front is rotated in towards the vehicle body.

Creating and Adjusting Subsystems

3. Select **pvl_toe_angle**. Note that it currently has a value of 0.0.
4. To see the effects of toe angle, in the **Real** text box, enter **2.0**.
Adams/Car updates both sides because Symmetric is set to yes.
5. Select **OK**.

A construction frame, cf[l,r]_wheel_center (a marker) in the model is parameterized on this parameter value and updates automatically. This will be discussed in more detail in subsequent chapters.



For related information, see the tabs Templates and Analyze in the Adams/Car online help.


Note: You can also adjust toe and camber angle by selecting
Adjust → Suspension Parameters → Toe/Camber Values.

Creating and Adjusting Subsystems

- **To adjust the springs:**

1. From the Model Browser, point to **Force Elements > Springs**, right click the spring you want to modify, select **Modify**. Or, right-click a spring in the model, and then select **Modify**.

Here you can adjust the property file that defines the force versus deflection, as well as either the installed length or the amount of preload, since these two quantities are directly related. The property file <acar_shared>/springs.tbl/mdi_0001.spr is already in the Property File text box.

2. To see the force-deflection curve, select the **Curve Manager tool** . The Curve Manager replaces your Standard Interface session. Notice that the spring is basically a piece-wise linear spring, with two spring rates.

Creating and Adjusting Subsystems

3. To return to Adams/Car Standard Interface, from the **File** menu, select **Close**.
4. To change the property file, right-click the **Property File** text box, point to **Search**, and then select **<acar_shared>\springs.tbl\MDI_125_300.spr**.
5. To see the force-deflection curve, select the Curve Manager tool again. Notice that this spring is a linear spring.
6. Return to Adams/Car Standard Interface (File → Close).
7. Change the installed length to **200**.
8. Select **OK**.

For a spring definition, see Springs, in Section 11.

Creating and Adjusting Subsystems

- **To replace the dampers:**

- Use the Replace feature to change the damper instance, that is, the type of damper.

1. From the Model Browser, point to **Force Elements>Dampers**, right click the damper you want to modify, select **Modify**. Or, right-click a damper, and then select **Modify**.

Here you can adjust the property file that defines the force versus velocity. The property file **<acar_shared>/dampers.tbl/mdi_0001.dpr** is already in the Property File text box.

2. To see the force-velocity curve, select the **Curve Manager** tool. Notice that it is a nonlinear curve.
3. Return to Adams/Car Standard Interface (**File → Close**).
4. Select **OK**.

Creating and Adjusting Subsystems

5. Right-click the damper and select **Replace**.
6. Set **Definition** to **Linear_damper**.
7. Set **Replace** to **All instances of the same name**.
8. Select **OK**.

The Modify Linear Damper dialog box appears.

9. Select the **View File** tool .

Note that the damping is set to a constant value in mdi_0001.lbf.

- **Reminder**

All of the modifications you just made in Standard Interface only to change your subsystem. There are many ways to perform these modifications. Note that the adjustments you make, will change your subsystem file in your Adams/Car database, only after you save it.

WORKSHOP 4

RUNNING SUSPENSION ANALYSIS

Running Suspension Analysis

- **Problem statement**

- In this workshop, **you will analyze and modify an assembly** of a front suspension and steering subsystem for **quasi-static simulation**
- You will learn how to carry out **Dynamic Suspension Analysis**
- You will **also create plot configuration file** and use it to plot curves in the Adams/Postprocessor
- The workshop contains the following sections:
 - To Run Quasi-Static Suspension Analysis
 - To Run Dynamic Suspension Analysis
 - To Create Plot Configuration File



This workshop takes about one hour to complete.

Running Suspension Analysis

- To Run Quasi-Static Suspension Analysis:

Go through Adams Online help → Getting Started → Getting Started Using Adams Car → Suspension Analysis Tutorial.

Note: You need to select **steering_wheel_input** to create plot in the Adams/PostProcessor. This characteristic is beneath **+testrig** when you switch to Adams/PostProcessor. Likewise, when you want to plot **scrub_radius**, you can find that beneath **+testrig**.

Running Suspension Analysis

- **To Run Dynamic Suspension Analysis:**
 - This allows you to directly provide a RPCIII file or define View Functions to specify Jack and Steering motion as a function of displacement, force, etc.
 - In this workshop, a Suspension Assembly consisting of a double wishbone suspension and a rack and pinion steering system. The dynamic suspension analysis is carried out to actuate the wheel pads across a range of frequencies.
 - We will look at the lower control arm bushing force in the Adams/Postprocessor.

Running Suspension Analysis

- **To Run Dynamic Suspension Analysis (Cont.):**

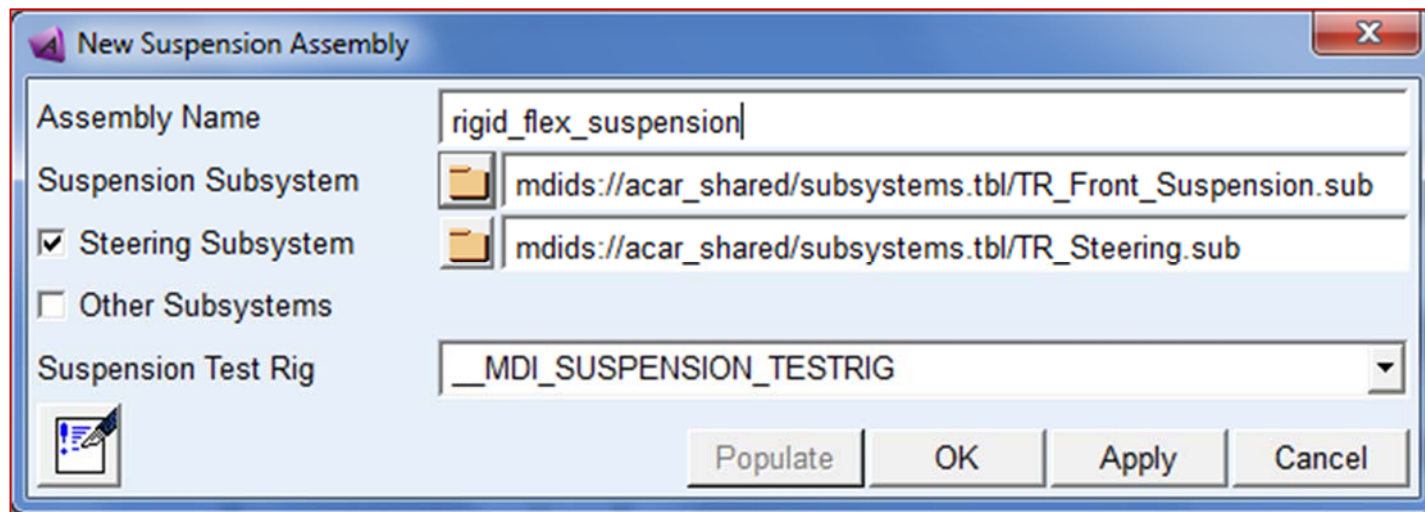
- Here is the procedure to analyze a double wishbone suspension with rigid lower control arm.

- Start Adams/Car, Select **Standard Interface**.

- Create a new Suspension assembly:

- **File → New → Suspension Assembly**

- Fill the dialog box as indicated below. To select the subsystems, Right-click – Search - **<acar_shared>** to open the file browser.



Running Suspension Analysis

- **To Run Dynamic Suspension Analysis (Cont.):**
 - The suspension assembly should be displayed.
 - To simulate using the Dynamic Solver Statements, go to **Simulate > Suspension Analysis > Dynamic**

Running Suspension Analysis

- To Run Dynamic Suspension Analysis (Cont.):

Suspension Analysis: Dynamic

Suspension Assembly: rigid_flex_suspension

Assembly Variant: default

Output Prefix: rigid_dynamic

Duration Time: 10

Number of Steps: 1000

Mode of Simulation: interactive

Vertical Setup Mode: Wheel Center

Coordinate System: ISO

Jack Excitation Mode: displacement

Vertical Input: arbitrary functions

Vertical Input Function:

Left Input	Right Input
STEP(time,2,10*sin(4*pi*time),8,30*sin(4*pi*time))	STEP(time,2,10*sin(4*pi*time),8,30*sin(4*pi*time))

Cornering Force: 0.0

Braking Force: 0.0

Traction Force: 0.0

Aligning Torque: 0.0

Overturning Torque: 0.0

Rolling Res. Torque: 0.0

Damage Force: 0.0

Damage Radius: 0.0

Steering Excitation Mode: displacement

Steering Motion: 0.0

☐ Compliance Matrix Requests

☒ Create Analysis Log File

OK Apply Cancel

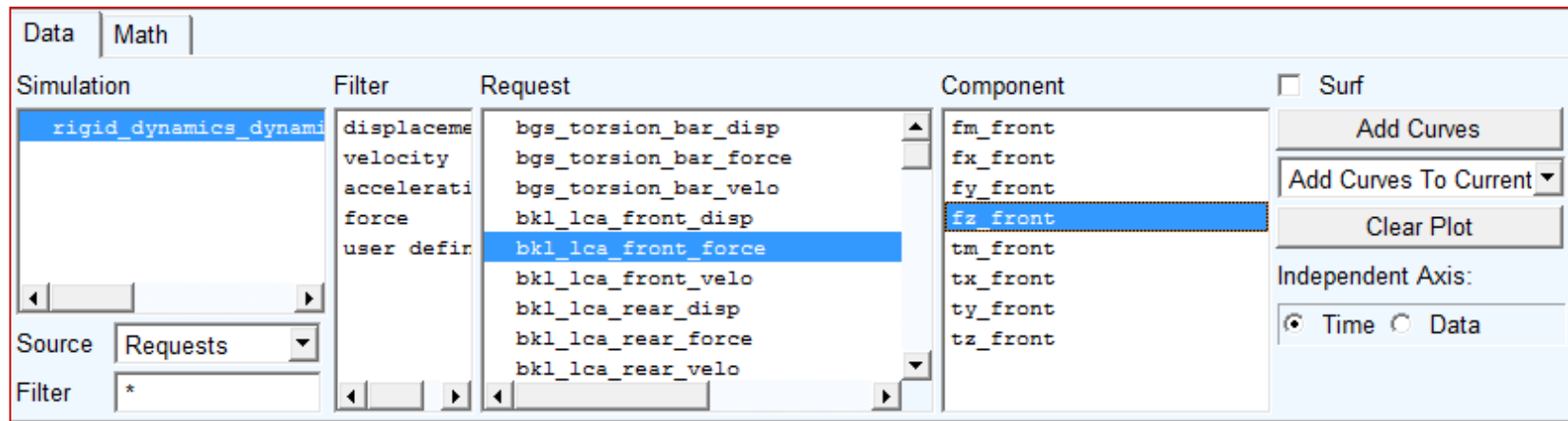
- Insert **step(time,2,10*sin(4*pi*time),8,30*sin(4*pi*time))** into both the left and right **Vertical Input Function** fields.

You will use Functions to define the vertical displacement of the Jack. The function steps up the amplitude from 10 mm in bounce and rebound to 30 mm respectively with a frequency of 4 Hz.

- Select **OK**.

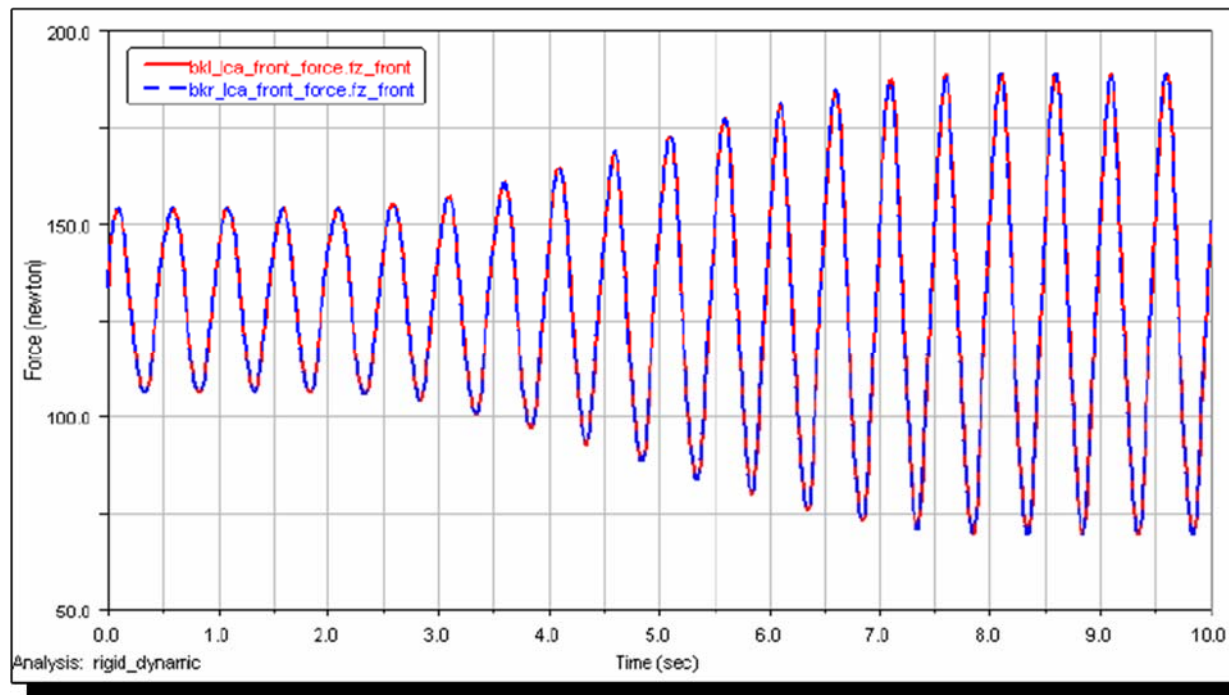
Running Suspension Analysis

- **To Run Dynamic Suspension Analysis (Cont.):**
 - Animate and Review the results:
 - From the **Review** menu, select **Animation Controls**
 - Animation Controls dialog box will be displayed. Animate the model and observe the change in the suspension travel.
 - Plot the bushing force in the lca_front bushing:
 - Hit the **F8** key in Adams/Car to switch to Adams/PostProcessor
 - Locate the **bkl_lca_front_force** and **bkr_lca_front_force** REQUEST under user-defined REQUESTs:



Running Suspension Analysis

- **Plot the bushing force in the lca_front bushing (Cont.):**
 - The fz_front component corresponds to magnitude of the force in the Z direction. Plot this quantity to obtain a figure similar to the one below:



Running Suspension Analysis

- **To Run Dynamic Suspension Analysis (Cont.):**

Optional task:

- You would like to go through the below SimCompanion article to see how the **force changes by replacing the rigid lower control arm by a flexible body**. In addition, you can get to know how to use the flex body swap dialog box to switch a rigid lower control arm with a flexible one.

Note: The files needed for this task are provided in the **acar_training_MD.cdb** database. You need not add the database provided in the article separately.

Using the Dynamic Suspension Testrig

Simcompanion article #: KB8017768

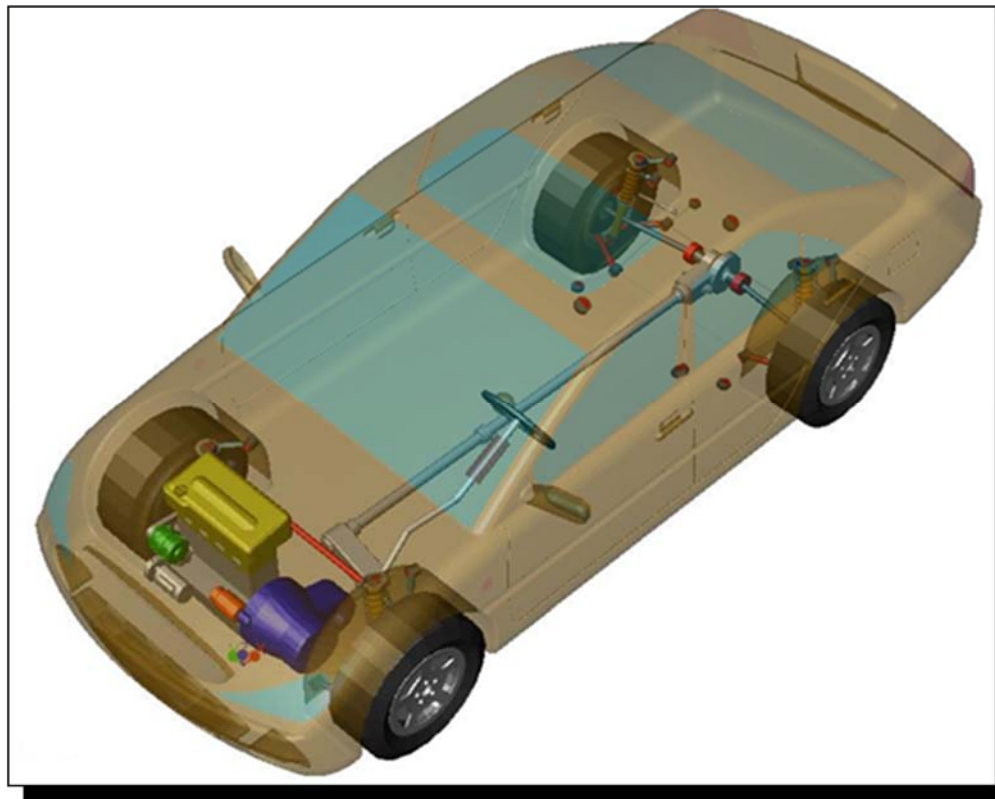
<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8017768>

Running Suspension Analysis

- **To Create and Run Plot Configuration File:**
 - Basic plot configuration steps:
 - Run a suspension analysis in the Standard Interface.
 - Manually make plots in Adams/PostProcessor.
 - From the **File** menu, point to **Export**, and then select **Plot Configuration File**, to save the plot configuration file (.plt) in plot_configs.tbl in your default writable database.
 - Re-run the analysis in Standard Interface.
 - From the **Plot** menu, select **Create Plots**, to load the plot configuration file (.plt) you just created in Adams/PostProcessor.
 - The same plots are made automatically with the plot configuration file.

WORKSHOP 5

IMPORTING CAD GEOMETRY



Importing CAD geometry

- **Problem Statement:**

In this workshop, you will learn how to import the CAD geometry in Adams/Car.

The workshop contains the following sections:

1. Attaching graphics shell (*.shl) file to the part



This workshop takes one hour to complete.

Importing CAD geometry

- **Attaching graphics shell (*.shl) file to the part:**
 - To use a shell file for adding graphics (geometry) to the part, the recommended approach **is to first create a part through the Build Menu.** Then add the geometry as the next step. (This is a little different from Adams/View in which you let Adams/View import the geometry and create the part in a single step.)

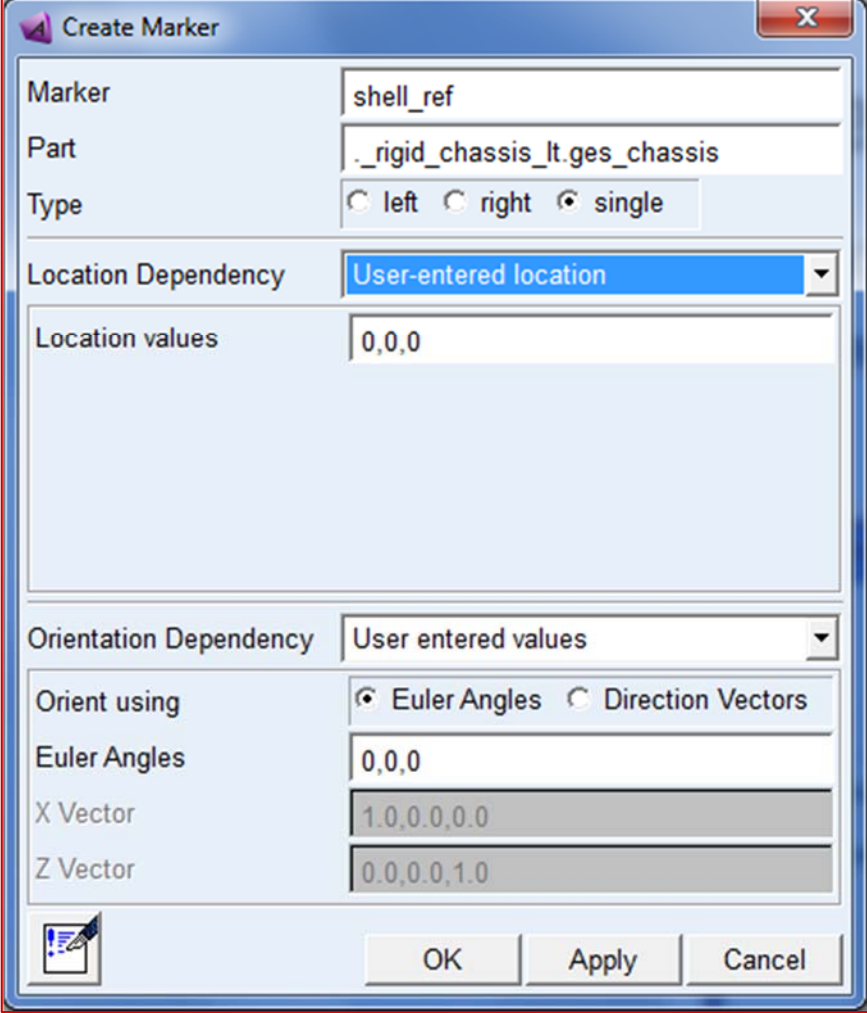
Note: The graphics shell file and any other geometry in different CAD format would not be parameterized.

Importing CAD geometry

- **Attaching graphics shell (*.shl) file to the part (Cont.):**
 - The shell file first needs to be saved in the database under the shell_graphics.tbl folder. Once you save the *.shl file in the database this file is ready to be imported in Adams/Car. For this workshop, the required shell file is already provided in the database - **acar_training_MD.cdb\shell_graphics.tbl**
- **To import CAD graphics:**
 - Open Adams/Car
 - Click on **Template Builder mode**
 - Select **File → Open** and in the Template name dialog box right click to point to **<acar_shared>/templates.tbl**
 - Select the template **rigid_chassis_lt.tpl** and click **OK**
 - Create a marker belonging to the part to which you want to attach the geometry
 - To create a marker, Go to **Build → Marker → New**

Importing CAD geometry

- To import CAD graphics (Cont.):
 - Enter the details as shown in the Create Marker dialog box.
 - Click **OK**



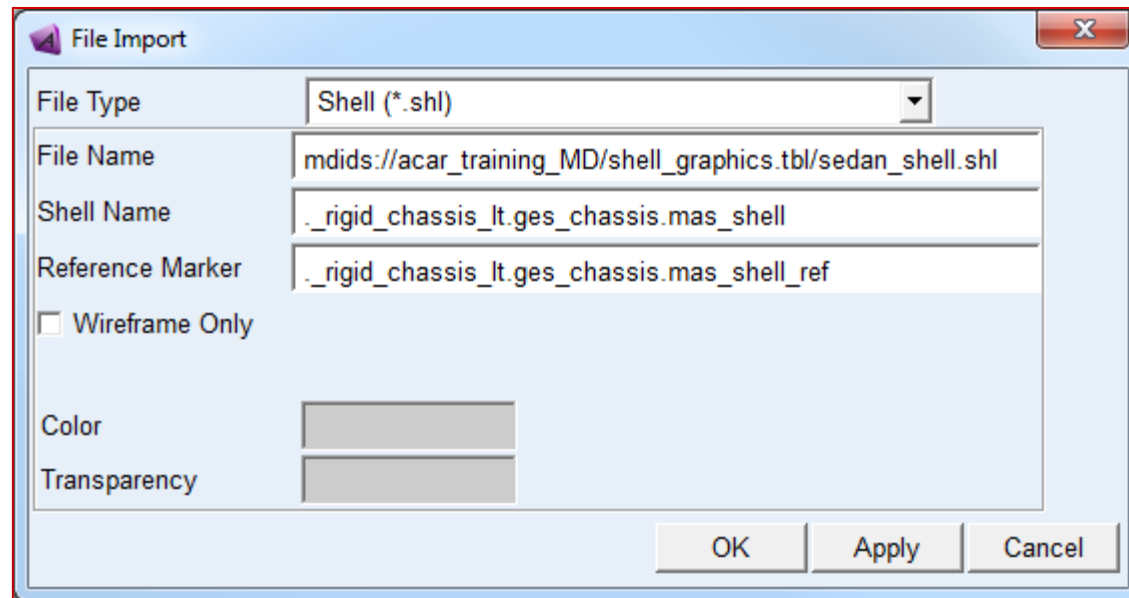
The image shows a 'Create Marker' dialog box with the following fields and options:

Marker	shell_ref
Part	._rigid_chassis_lt.ges_chassis
Type	<input type="radio"/> left <input type="radio"/> right <input checked="" type="radio"/> single
Location Dependency	User-entered location
Location values	0,0,0
Orientation Dependency	User entered values
Orient using	<input checked="" type="radio"/> Euler Angles <input type="radio"/> Direction Vectors
Euler Angles	0,0,0
X Vector	1.0,0.0,0.0
Z Vector	0.0,0.0,1.0

Buttons: OK, Apply, Cancel

Importing CAD geometry

- To import CAD graphics (Cont.):
 - Go to **File** → **Import** → **Shell**
 - The dialog box shown would pop-up. Enter the details as shown in the dialog box.
 - Click **OK**



Note:

Since the reference marker and shell graphic belong to the same part, the easiest way to fill out this dialog box is to first use **Pick** to select the **Reference Marker** you just created, and copy/paste the part name **._rigid_chassis_lt.ges_chassis** followed by the name of the shell, in this case, "body_shell".

Importing CAD geometry

- **To import CAD graphics (Cont.):**

- **Note:**

- The procedure remains similar when you want to import CAD geometry other than shell format.
 - The CAD formats like IGES (*.igs) and Parasolids (*.xmt_txt) can be imported in the similar fashion by specifying the part name as `._rigid_chassis_lt.ges_chassis`.
 - You can however use a shell file that is generated from these IGES and Parasolid formats.
 - The *.igs and *.xmt_txt can be imported into Adams using option (File → Import) and that could be exported as a Shell geometry (File → Export → Shell) for further use in Adams/Car.

WORKSHOP 6

RUNNING FULL-VEHICLE ANALYSIS & ADJUSTING MASS

Running Full-Vehicle Analysis

- **Problem statement**

- In this workshop:

- You will learn how to create a full-vehicle assembly, run different types of analyses, and view the results using animation and plotting.
 - Adjust the mass properties of an assembly to match the mass properties you input
 - Perform a Static Vehicle Setup analysis and generate vehicle setup report



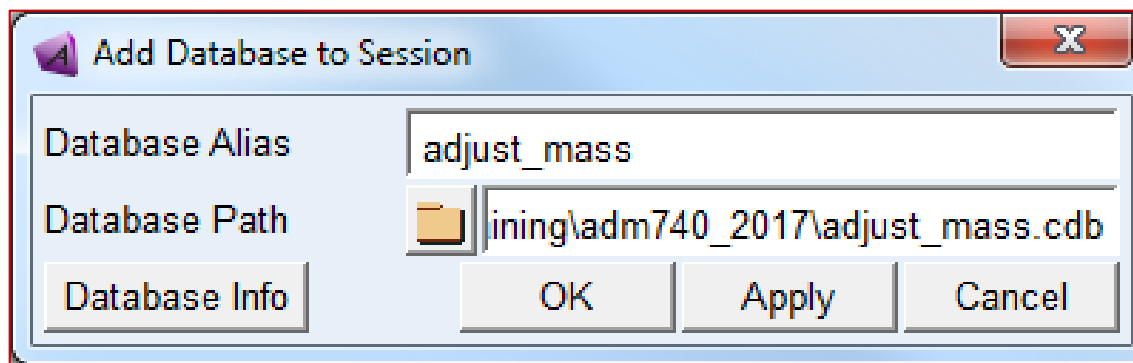
This workshop takes about one hour to complete.

Running Full-Vehicle Analysis

- **To Run Full-Vehicle Analysis:**
 - Go through **Adams Online help → Getting Started → Getting Started Using Adams Car → Full-Vehicle Analysis Tutorial.**

Adjusting Mass

- **Add a new database to your session:**
 1. Start Adams Car **Standard Interface**.
 2. From the **Tools** menu, point to **Database Management**, and then select **Add to Session**.
 3. In the **Database Alias** text box, enter **adjust_mass**.
 4. In the **Database Path** field, click on the **Folder** icon and point to **adjust_mass.cdb** that the instructor provided.
 5. Click **OK**.



Adjusting Mass

- **Calculate Aggregate Mass of the Vehicle**
 1. Go to **File – Open – Assembly** and choose **adjust_mass_asy** from the **<adjust_mass>/assemblies.tbl/**
 2. This assembly already has the body CG defined by a hardpoint **HPS_Chassis_CG** and marker attached to the body for the inertia axes **mas_chassis_cg_marker**.
 3. Zoom in to the yellow ellipsoid and highlight both.
 4. Right click **HPS_Chassis_CG** and modify the location of the HPS to **1500, 0.0, 450**. Since the marker **mas_chassis_cg_marker** is parameterized to the HP, it moves as the HP is moved. (Alternatively, from the Pull Down menu located at the top of Model Browser, select **.adjust_mass_asy.TR_Body**, point to **Hardpoint** and then right click **hps_Chassis_CG**, select **Modify**. Enter the location of HP as **1500,0.0,450**)
 5. Go to **Tools – Aggregate Mass** and click **OK**. Note the CG location and inertia tensor with respect to global reference frame

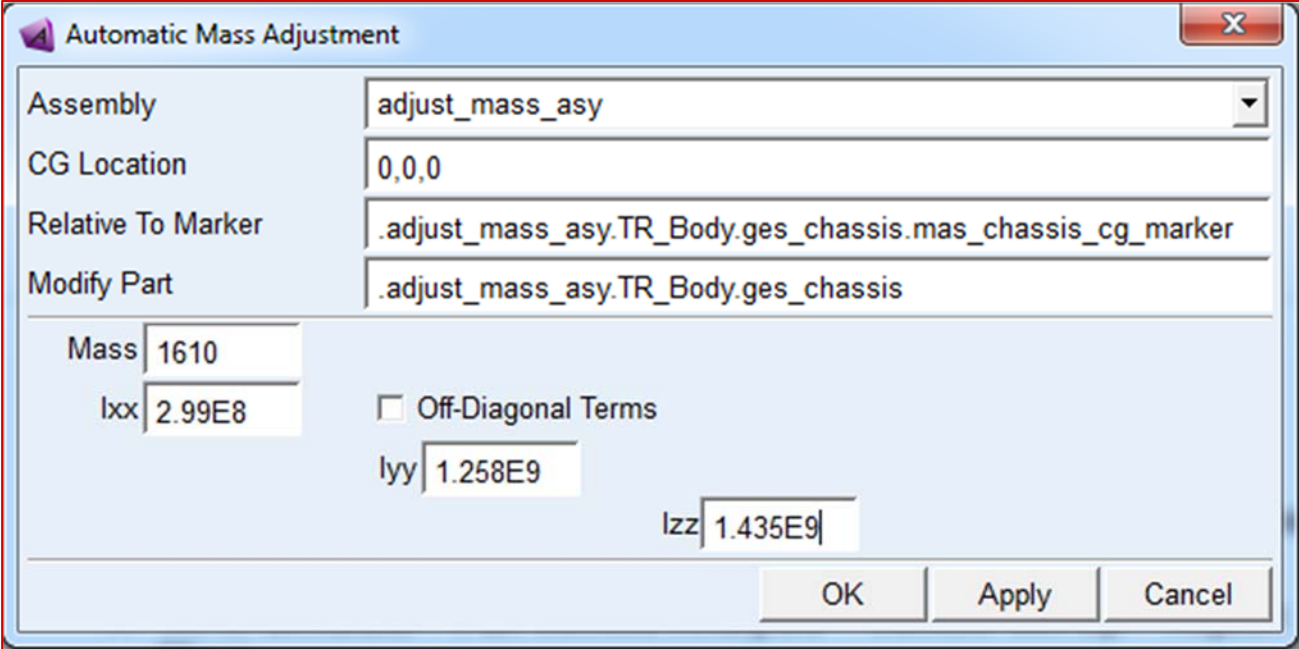
Adjusting Mass

- **Calculating Aggregate Mass of the Vehicle**
 6. Calculate **Aggregate Mass** making sure that in the **Relative To** option, pick the **mas_chassis_cg_marker** belonging to the chassis part.
 7. Compare the results between two results above.
 8. Also Right Click on **Chassis** part and select info and note the mass of the chassis part

Adjust Mass

- **Automatic Mass Adjustment**

- You can automatically adjust the mass properties of an assembly to match the mass properties you input.
1. Go to **Simulate – Full Vehicle Analysis – Vehicle Set-Up – Adjust Mass...**
 2. Fill the dialog box as shown next
 3. Compare the mass of chassis after this Mass Adjustment



The image shows a software dialog box titled "Automatic Mass Adjustment". It contains several input fields for configuring mass adjustment. The "Assembly" field is set to "adjust_mass_asy". The "CG Location" field is set to "0,0,0". The "Relative To Marker" field is set to ".adjust_mass_asy.TR_Body.ges_chassis.mas_chassis_cg_marker". The "Modify Part" field is set to ".adjust_mass_asy.TR_Body.ges_chassis". Below these fields, there are input boxes for "Mass" (1610), "Ixx" (2.99E8), "Iyy" (1.258E9), and "Izz" (1.435E9). There is an unchecked checkbox labeled "Off-Diagonal Terms". At the bottom right, there are three buttons: "OK", "Apply", and "Cancel".

Assembly	adjust_mass_asy
CG Location	0,0,0
Relative To Marker	.adjust_mass_asy.TR_Body.ges_chassis.mas_chassis_cg_marker
Modify Part	.adjust_mass_asy.TR_Body.ges_chassis
Mass	1610
Ixx	2.99E8
Iyy	1.258E9
Izz	1.435E9
Off-Diagonal Terms	<input type="checkbox"/>

Static Vehicle Set-up

- Using Static Vehicle Set-Up analysis you can set realistic suspension alignment, cross-weight and ride height adjustment for your vehicle.
- To run the static vehicle set-up analysis:
 1. Open the **MDI_Demo_Vehicle** from the **acar_shared** database in the standard interface of Adams Car.
 2. From the **Simulate** menu, point to **Full-Vehicle Analysis** and then select **Vehicle Set-Up > Static Vehicle Set-Up**.

The following dialog box will appear:

- **Select the 2 actions to be performed with the Vehicle**
 1. **Save-** Export the current dialog box contents into a vehicle set-up file
 2. **Load-** Reads the Set-up definition file and populates the simulation dialog box accordingly

The screenshot shows the 'Static Vehicle Set-Up' dialog box. It contains the following fields and options:

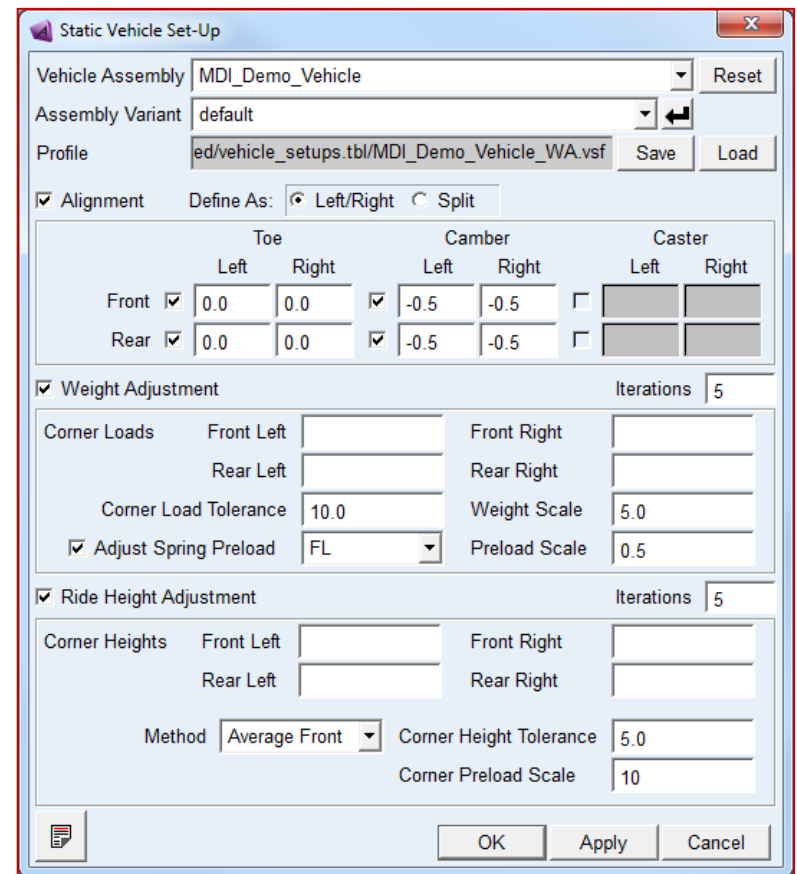
- Vehicle Assembly:** MDI_Demo_Vehicle (with a Reset button)
- Assembly Variant:** default (with a dropdown arrow)
- Profile:** (empty field with Save and Load buttons)
- Alignment:** ☒ (Define As: Left/Right, Split)
- Toe:** Front Left: 0.0, Right: 0.0; Rear Left: 0.0, Right: 0.0
- Camber:** Front Left: -0.5, Right: -0.5; Rear Left: -0.5, Right: -0.5
- Caster:** Front Left: (empty), Right: (empty); Rear Left: (empty), Right: (empty)
- Weight Adjustment:** ☒ (Iterations: 5)
 - Corner Loads: Front Left, Front Right, Rear Left, Rear Right (all empty)
 - Corner Load Tolerance: 10.0
 - Weight Scale: 5.0
 - ☒ Adjust Spring Preload: FL (dropdown), Preload Scale: 0.5
- Ride Height Adjustment:** ☒ (Iterations: 5)
 - Corner Heights: Front Left, Front Right, Rear Left, Rear Right (all empty)
 - Method: Average Front (dropdown)
 - Corner Height Tolerance: 5.0
 - Corner Preload Scale: 10
- Buttons:** OK, Apply, Cancel

Static Vehicle Set-up

- Right click the space next to **Profile** and select the vehicle set-up definition file **MDI_Demo_Vehicle_WA.vsf** from **acar_shared** database, press **Apply** button next to it to populate the weight adjustment parameters, as shown in the following dialog box:

Note:


You can select single or all the three adjustments at a time and run an analysis.

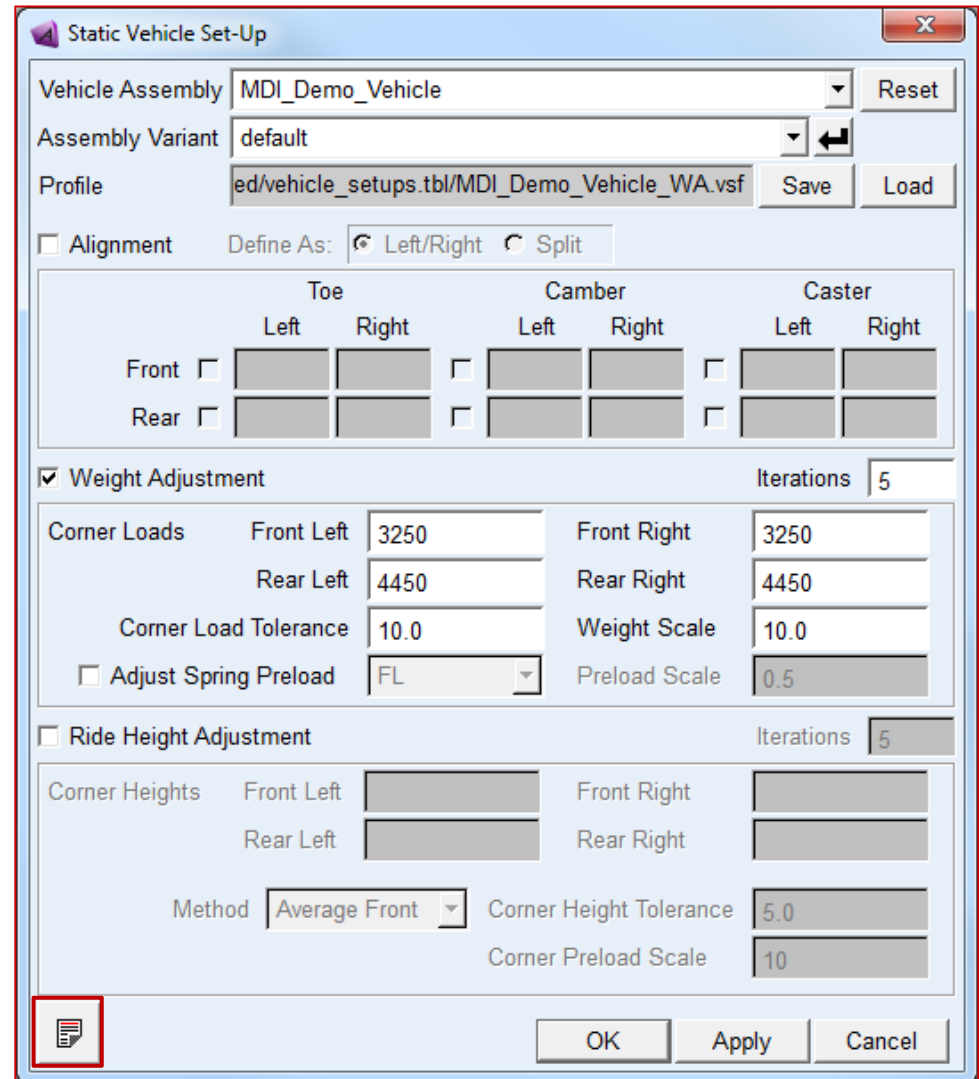


The dialog box is titled "Static Vehicle Set-Up". It contains the following fields and controls:

- Vehicle Assembly:** MDI_Demo_Vehicle (dropdown menu) with a **Reset** button.
- Assembly Variant:** default (dropdown menu) with a **Reset** button.
- Profile:** ed/vehicle_setups.tbl/MDI_Demo_Vehicle_WA.vsf (text field) with **Save** and **Load** buttons.
- Alignment:** ☒ **Define As:** ☒ Left/Right ☐ Split
- Toe:** Front Left: 0.0, Front Right: 0.0, Rear Left: 0.0, Rear Right: 0.0. ☒ for each.
- Camber:** Front Left: -0.5, Front Right: -0.5, Rear Left: -0.5, Rear Right: -0.5. ☒ for each.
- Caster:** Front Left: [blank], Front Right: [blank], Rear Left: [blank], Rear Right: [blank]. ☐ for each.
- Weight Adjustment:** ☒ **Iterations:** 5. **Corner Loads:** Front Left, Front Right, Rear Left, Rear Right. **Corner Load Tolerance:** 10.0. **Adjust Spring Preload:** ☒ **FL** (dropdown). **Weight Scale:** 5.0. **Preload Scale:** 0.5.
- Ride Height Adjustment:** ☒ **Iterations:** 5. **Corner Heights:** Front Left, Front Right, Rear Left, Rear Right. **Method:** Average Front (dropdown). **Corner Height Tolerance:** 5.0. **Corner Preload Scale:** 10.
- Buttons:** **OK**, **Apply**, **Cancel**.

Static Vehicle Set-up

- To run the selected analysis press **Apply** tab at the right bottom corner of the dialog box.
- After finishing the analysis small window will pop-up showing information about weight distribution over all four wheels, press **OK**, close the message window.
- To see the result of analysis press the  (**Show Report**) tab.



The image shows the 'Static Vehicle Set-Up' dialog box. It contains several sections for configuring vehicle analysis parameters. At the top, there are dropdown menus for 'Vehicle Assembly' (MDI_Demo_Vehicle), 'Assembly Variant' (default), and 'Profile' (ed/vehicle_setups.tbl/MDI_Demo_Vehicle_WA.vsf), along with 'Reset', 'Save', and 'Load' buttons. Below this is an 'Alignment' section with a 'Define As' dropdown set to 'Left/Right'. The main body is divided into three sections: 'Toe', 'Camber', and 'Caster', each with input fields for 'Left' and 'Right' for both 'Front' and 'Rear' wheels. The 'Weight Adjustment' section is checked and includes 'Corner Loads' (Front Left: 3250, Rear Left: 4450, Front Right: 3250, Rear Right: 4450), 'Corner Load Tolerance' (10.0), 'Adjust Spring Preload' (FL), 'Iterations' (5), 'Weight Scale' (10.0), and 'Preload Scale' (0.5). The 'Ride Height Adjustment' section is unchecked and includes 'Corner Heights' (Front Left, Rear Left, Front Right, Rear Right), 'Method' (Average Front), 'Corner Height Tolerance' (5.0), and 'Corner Preload Scale' (10). At the bottom, there is a 'Show Report' icon (a document with a magnifying glass) and 'OK', 'Apply', and 'Cancel' buttons.

Toe		Camber		Caster		
	Left	Right	Left	Right	Left	Right
Front	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Rear	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Weight Adjustment		Iterations
Corner Loads	Front Left: 3250, Rear Left: 4450, Front Right: 3250, Rear Right: 4450	5
Corner Load Tolerance	10.0	
Adjust Spring Preload	FL	
Weight Scale	10.0	
Preload Scale	0.5	

Ride Height Adjustment		Iterations
Corner Heights	Front Left, Rear Left, Front Right, Rear Right	5
Method	Average Front	
Corner Height Tolerance	5.0	
Corner Preload Scale	10	

Static Vehicle Set-up

- The following window will display the result of the analysis:

Static Vehicle Set-Up Report

VEHICLE SET-UP REPORT

27 Sep 2017 12:15

VEHICLE ASSEMBLY:
<acar_shared>/assemblies.tbl/MDI_Demo_Vehicle.asy

(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	15400.0	7700.0	7700.0
	%		50.0	50.0
Front ground reaction	N [%]	6500.0 [42.2]	3250.0	3250.0
Rear ground reaction	N [%]	8900.0 [57.8]	4450.0	4450.0
Total roll inertia	kg-mm**2	2.9883386e+08		
Total pitch inertia	kg-mm**2	1.1625523e+09		
Total yaw inertia	kg-mm**2	1.3412270e+09		
Total product Ixy	kg-mm**2	1.6220169e+06		
Total product Ixz	kg-mm**2	-5.1450943e+06		
Total product Iyz	kg-mm**2	-3.6730867e+05		
Global C.G. location	mm	414.83		
Ground plane elevation	mm	16.22		
C.G. height	mm	398.60		
Body yaw angle	deg	0.000		
Body pitch angle	deg	-0.551		
Body roll angle	deg	0.000		
Wheelbase	mm	2560.18	2560.18	2560.18
Average track width	mm	1558.99		
Static stability factor	-	1.956		

FRONT AXLE:
<acar_shared>/subsystems.tbl/TR_Front_Suspension.sub
<acar_shared>/tires.tbl/TR_front_pac89.tir
<acar_shared>/tires.tbl/TR_front_pac89.tir

(PARAMETER)	(UNITS)	(AVERAGE)	(LEFT)	(RIGHT)
-------------	---------	-----------	--------	---------

Close



WORKSHOP 7

CREATING EVENT FILES

Event Builder

- **Problem Statement:**
 - In this workshop, you will learn the basics of **creating multiple mini-maneuvers**, establishing the appropriate application areas for those mini-maneuvers, and settings end conditions for each mini-maneuver.
- **The example contains the following sections:**
 - Creating an Event
 - Creating the Three Mini-Maneuvers
 - Defining the Mini-Maneuver Parameters
 - Running the New Analysis



This workshop takes one hour to complete.

Event Builder

- **Creating an Event**

- Before you can create an event, you must create a new file, as explained next. The instructions will point out the differences between procedures in Adams/Car.

- **To display the Event Builder:**

- Open the assembly **MDI_Demo_Vehicle** from the **acar_shared** database.
- From the **Simulate** menu, point to **Full-Vehicle Analysis**, and then select **Event Builder**.
- Adams displays the Event Builder, which is grayed out because you do not have a .xml file yet.

Event Builder

- **To create a new file:**

1. From the Event Builder's **File** menu, select **New**.

The New File dialog box appears.

2. Enter the name of the event: **braking_in_a_turn**. This is the name of the file that will be saved to your hard drive or network drive with a .xml extension.

3. Select **OK**.

Your Adams product creates the file and also defines the first mini-maneuver, MINI_1, with the following specifications:

- Active: yes
- Abort Time: 10.0
- Step Size: 0.01
- Sample Period: 0.01

Event Builder

- **Creating the Three Mini-Maneuvers**

- To complete the braking-in-turn event, you must create three mini-manuevers.

- **To create the mini-manuevers:**

1. To go to the mini-maneuver creation interface, select .

The mini-maneuver creation interface appears. Note that by default, the first minimaneuver, MINI_1, is already defined.

2. To create one more mini-maneuver, in the **Name** text box, enter **MINI_2**, and then select **Add**.
3. To create the last mini-maneuver, in the **Name** text box, enter **MINI_3**, and then select **Add**.

The mini-maneuver window appears as follows:

Name	Active	Abort Time	Step Size	Sample Period
MINI_1	yes	10.0	0.01	0.01
MINI_2	yes	10.0	0.01	0.01
MINI_3	yes	10.0	0.01	0.01

Event Builder

- You can specify the behavior of the gear shifting parameters, but for this exercise, the defaults should be adequate. You can also specify the static setup for the first mini-maneuver, as well as the initial values for gear position and speed, as explained next.
 - Because the first mini-maneuver that you specify is going to be a straight line, **you will use the straight static setup**. For a description of the static setup methods available in the Driving Machine, see the Working with the Driver Control Data file page in Using Driving Machine section under Running Analyses of the Adams/Car online help.
4. To change the static setup to straight, select the **Static Set-up** tab, and then set **Task** to **straight**. Note that the rest of the parameters are not used because they will be overwritten by the initial values computed during the quasi-static setup.
- You must set the initial velocity of the vehicle. The value of the initial velocity depends on the units in which you are using the Event Builder.

Event Builder

5. To check the units or make a change, from the **Settings** menu (the menu within the Event Builder, not the system setting), select **Units**.
By default, the length unit is set to meters and the time unit is set to seconds. Therefore, the velocity must be set in meters/second.
6. To verify that velocity units are in meters/second, click in the **Speed** text box at the top of the Event Builder, and then read what is displayed in the **Current Field Unit** text box at the bottom of the Event Builder.
7. To set the target vehicle velocity, in the **Speed** text box, enter **27.7** (as previously explained, units are meters/second), which corresponds to approximately 100 km/h.
8. To set the initial gear, in the **Gear** text box next to the **Speed** box, enter **5**.
9. In the **Abort Time** for **MINI_2**, change the abort time to **5**.
10. To save the mini-maneuver, select **Save**.

Your Adams product saves the file, braking_in_a_turn.xml, to your current working directory.

Event Builder

- **Defining the Mini-Maneuver Parameters**
 - Now that you defined the three mini-maneuvers, you must define what each of the application areas and end conditions will do during the mini-maneuvers.
 - The application areas are steering, throttle, brake, clutch, and gear. For each application area, you define the Driving Machine control (open loop, closed loop, or human).
 - The end conditions define a vehicle state by which the mini-maneuver will be terminated. End conditions are extremely flexible and can be used for many purposes. The following are some example end conditions:
 - End conditions can have a single condition, such as time is greater than 5 seconds.
 - End conditions can have a single value that must be within a tolerance over a prescribed period of time. Lateral acceleration is 0.5 g over a two-second period and within a 0.05 g tolerance.

Event Builder

- Multiple end conditions, such as time is greater than 5 seconds or vehicle velocity is lower than 10 kph.
 - Combined end condition, such as vehicle velocity equal 100 kph (within a tolerance) and longitudinal acceleration equals 0 g (within a tolerance). The grouping of end conditions is permitted, allowing multiple conditions to be satisfied before a maneuver can be terminated.
- For more information on end conditions, see the Driving Machine section in the Adams/Car online help.

Event Builder

- **To define mini-maneuver parameters for MINI_1:**
 1. To access the mini-maneuver editor, double-click the name of the mini-maneuver you want to edit. In this case, **MINI_1**.
 2. For the first mini-maneuver, the vehicle should drive in a straight line and maintain constant velocity. To achieve this, set the following parameters:
 - **Steering** tab
 - **Actuator Type:** torque
 - **Control Method:** machine
 - **Control Mode:** Absolute
 - **Steer Control:** straight

The Steer Control setting enables the Driving Machine to drive the vehicle in a straight line.

Event Builder

- **Throttle tab**

- **Control Method:** machine
- **Control Mode:** Absolute
- **Speed Control:** maintain
- **Velocity:** 27.7

The Velocity setting enables the Driving Machine to maintain the steady-state velocity that you established earlier. The throttle will be controlled to maintain this velocity much like a cruise control system.

- **Braking tab**

- **Control Method:** machine

The Throttle and Brake settings control the longitudinal velocity of the vehicle.

- **Gear tab**

- **Control Method:** open
- **Control Type:** constant
- **Control Value:** 5

Maintain the same gear specified for the static setup.

- **Clutch tab**


- **Control Method:** open
- **Control Type:** constant
- **Control Value:** 0.0

Event Builder

- For the first mini-maneuver, you're using the Driving Machine for the steering and the throttle so you can maintain the vehicle speed during straight-line driving.
- The purpose of the first mini-maneuver is to **reach a dynamic steady-state end condition**. Although not absolutely necessary for this type of event, it helps demonstrate some important aspects of the Driving Machine. To satisfy a steady-state condition, the vehicle must be traveling a straight line (very low lateral acceleration) at a stable velocity (very low longitudinal acceleration). You can use end conditions to group these two conditions together to satisfy the steady-state requirements.

Event Builder

- To create end conditions for MINI_1:

1. Select the **Conditions** tab.
2. Select .
3. In the **Name** text box, enter **END_2**.
4. Select **Add**.
5. Modify the end conditions as follows:

- **END_1**

- **Condition Type:** Ion_accel
- **Test Type:** ==
- **Trigger Value:** 0.0
- **Error:** 0.05
- **Filter Time:** 2.0
- **Delay Time:** 0.0
- **Group Name:** MINI_1


Event Builder

- **END_2**
 - **Condition Type:** lat_accel
 - **Test Type:** ==
 - **Trigger Value:** 0.0
 - **Error:** 0.05
 - **Filter Time:** 2.0
 - **Delay Time:** 0.0
 - **Group Name:** MINI_1

Note that you used MINI_1 for both group names. Because END_1 and END_2 share the same name, both end conditions must be satisfied for the maneuver switch to occur.

6. Select **Save**.

Event Builder

- To define mini-maneuver parameters for **MINI_2**:
 1. Select .
 2. Double-click on the name of the mini-maneuver, **MINI_2**.
 3. Set the mini-maneuver parameters as follows:
 - **Steering** tab
 - **Actuator:** torque
 - **Control Method:** machine
 - **Control Mode:** Absolute
 - **Steer Control:** skidpad
 - **Entry Distance:** 10.0
 - **Radius:** 120
 - **Turn Direction:** Right

These parameters will start the vehicle turning at the beginning of the second mini-maneuver.

Event Builder

- **Throttle tab**

- **Control Method:** machine
- **Control Mode:** Absolute
- **Speed Control:** maintain
- **Velocity:** 27.7

These parameters allow the Driving Machine to maintain the straight line velocity established in MINI_1. The steering parameters established in the steering block are not modified.

- **Braking tab**

- **Control Method:** machine

- **Gear tab**

- **Control Method:** open
- **Control Type:** constant
- **Control Value:** 5

Maintain the same gear specified for the static setup.

Event Builder

- **Clutch** tab
 - **Control Method:** open
 - **Control Type:** constant
 - **Control Value:** 0.0

For the second mini-maneuver, you're using the Driving Machine for the steering and the throttle so you can make a right turn and control the vehicle on a radius of 120 m.

Next, you will create two end conditions to verify that a steady-state cornering condition has been achieved.

Event Builder

- **To create end conditions for MINI_2:**
 1. Select the **Conditions** tab.
 2. In the **Name** text box, enter **END_1**.
 3. Select **Add**.
 4. Create the second end condition, **END_2**, just as you created the first.
 5. Modify the end conditions as follows:
 - **END_1**
 - **Condition Type:** **Ion_accel**
 - **Test Type:** **==**
 - **Trigger Value:** **0.0**
 - **Error:** **0.05**
 - **Filter Time:** **2.0**
 - **Delay Time:** **0.0**
 - **Group Name:** **MINI_2**


Event Builder

- **END_2**
 - **Condition Type:** **curvature**
 - **Test Type:** **==**
 - **Trigger Value:** **0.00833**
 - **Error:** **0.00005**
 - **Filter Time:** **2.0**
 - **Delay Time:** **0.0**
 - **Group Name:** **MINI_2**

These end conditions ensure that a radius of approximately 120 m is followed at a velocity of 100 kph.

6. Select **Save**.

Event Builder

- To define mini-maneuver parameters for **MINI_3**:
 1. Select .
 2. Double-click on the name of the mini-maneuver, **MINI_3**.
 3. Set the mini-maneuver parameters as follows:
 - **Steering** tab
 - **Actuator:** torque
 - **Control Method:** machine
 - **Control Mode:** Absolute
 - **Steer Control:** skidpad
 - **Entry Distance:** 0.0
 - **Radius:** 120
 - **Turn Direction:** Right

These parameters maintain the radius from the previous mini-maneuver.
 - **Throttle** tab
 - **Control Method:** machine

Event Builder

- **Braking tab**

- **Control Method:** machine
- **Speed Control:** lon_accel
- **Start Time:** 1.0
- **Long. Acc.:** - 3.0

These parameters drop the throttle at the beginning of the third mini-maneuver to zero, and control the deceleration to 3.0 m/s².

- **Gear tab**

- **Control Method:** machine

The Driving Machine controls the gear selection.

- **Clutch tab**

- **Control Method:** machine

The Driving Machine controls the clutch selection.

For the third mini-maneuver, you use the Driving Machine for all vehicle activity; the Driving Machine will maintain the vehicle radius while at the same time brake the vehicle at 3 m/s². You will use a single end condition for the third mini-maneuver, which is velocity. You will trigger the end of the simulation if the velocity is below 10 kph which is approximately 2.77 m/s.

Event Builder

- **To create the end condition for MINI_3:**
 1. Select the **Conditions** tab.
 2. In the **Name** text box, enter **END_1**.
 3. Select **Add**.
 4. Modify the end condition as follows:
 - **END_1**
 - **Condition Type:** **velocity**
 - **Test Type:** **<<**
 - **Trigger Value:** **2.77**
 - **Error:** **0.01**
 - **Filter Time:** **0.0**
 - **Delay Time:** **0.0**

Event Builder

5. Select **Save**.

By default, Adams saves the file in the current working directory. If you are not sure where the current working directory is located, do following:

- In Adams/Car, from the **File** menu, select **Select Directory**.
- You find the directory location in the Browse For Folder dialog box; it may be similar to: C:\Documents and Settings\ user_name or /home/usr/username.

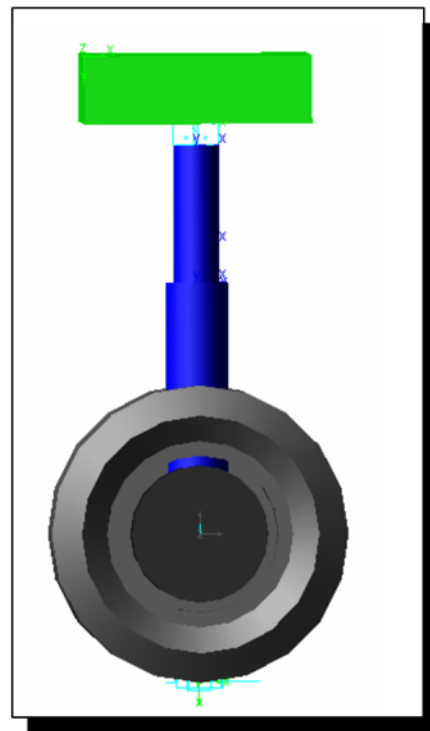
Event Builder

- **Running the New Analysis**

- To run the new analysis:
 - From the **Simulate** menu, point to **Full-Vehicle Analysis**, and then select **File Driven Events**.
 - In the **Driver Control Files** text box, enter the name of the new event file you created in this tutorial (**braking_in_a_turn.xml**).
 - Fill out the other parameters. For help on filling out this dialog box, press **F1**.
- You are now ready to use the Event Builder to run full-vehicle analyses.

WORKSHOP 8

TIRE TESTRIG TUTORIAL



Tire Testrig

- **Problem Statement:**

- In this workshop, you will learn how to use the new Tire Testrig available in Adams/Car.

- **The testrig provides the ability to visualize the effects of tire properties and compare tire model performance.**
- **The testrig provides the capability to easily produce plots of tire characteristics using a tire property file and a list of inputs.**



This workshop takes about one hour to complete.

Tire Testrig

- The tire testrig model consists of single tire (wheel) mounted on a spindle that can be suspended to ground by a spring, or preloaded by a single component force or fixed at a certain axle height. The road can be **fixed or moving** (moving in x, y, z-direction, or rotating around x-axis), while the road itself can be flat with or without a cleat (pothole), or user-defined (road property file).
- The wheel movements can enhance **steering** (slip angle variations), **wheel rotations** (longitudinal slip variations), **inclination** with the road (camber), **vertical and longitudinal wheel (center) displacements**.

Tire Testrig

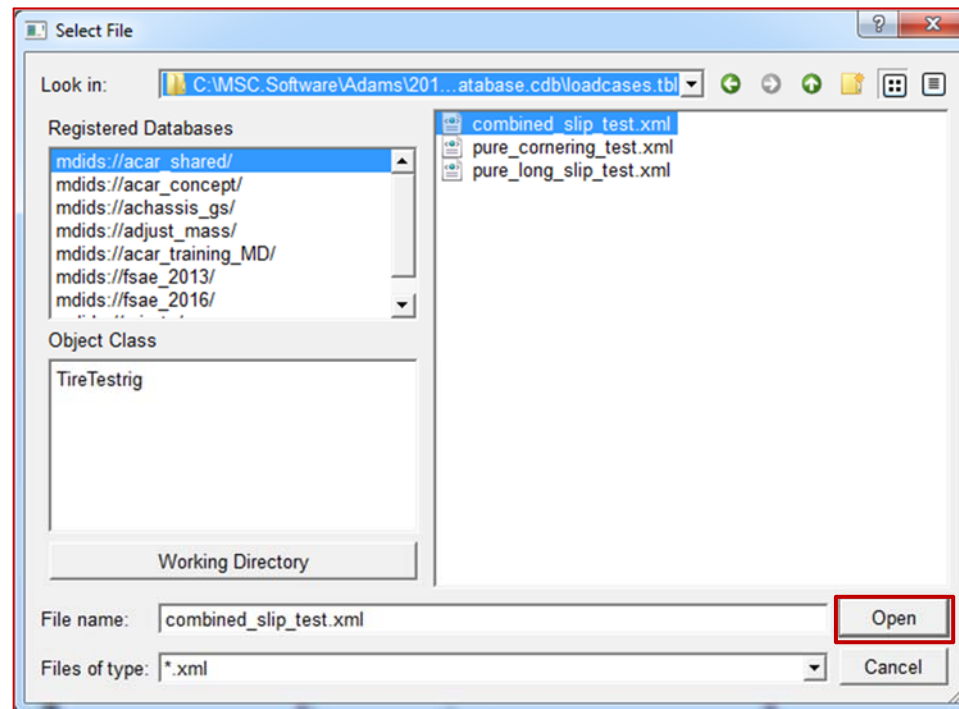
- **The procedures to use Tire Testrig:**
 - This workshop is split into two parts. One where the default loadcase file is being used and one where you create your own loadcase file.
 - **Using the shared tire loadcase file**
 - Run a loadcase file with the tire testrig
 - Reviewing the results in Adams/Postprocessor
 - **Creating a tire loadcase file**
 - Creating a tire loadcase file
 - Reviewing the results in Adams/Postprocessor

Tire Testrig

- **Using the shared tire loadcase file:**
 - In this section you will see how to use and modify the existing tire loadcase file. The default loadcase file (**combined_slip_test.xml**) has been provided in the loadcase.tbl under **acar_shared_database**.
 - This loadcase has four analyses which define four experiments for the tire testrig. The tire characteristics at different slip angles are measured. A longitudinal slip sweep is carried out at each of the slip angles of interest (0, 2, 5 and 10 degrees).

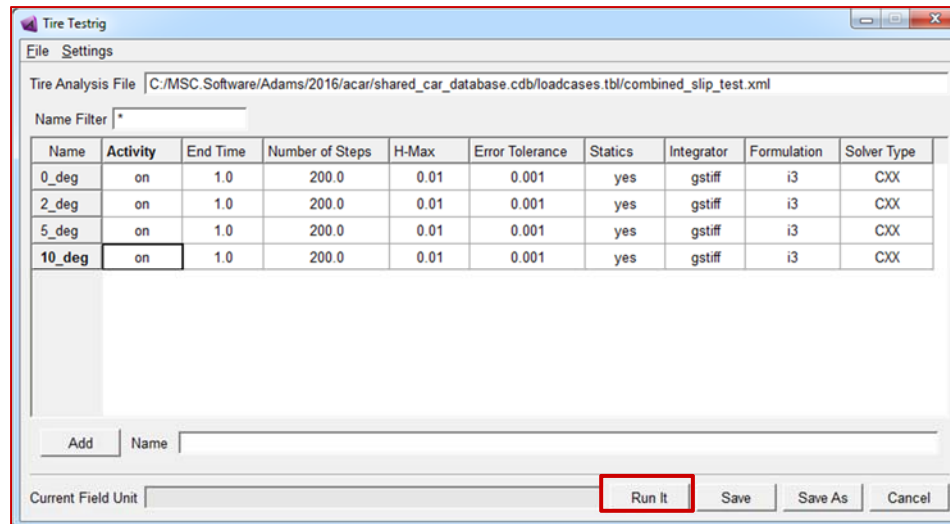
Tire Testrig

- Run a loadcase file with the tire testrig:
 1. **Open** Adams Car session.
 2. Click **Simulate** → **Component Analysis** → **Tire Testing**
Then go to **File** → **Open**
The **Tire Testrig** screen will appear
 3. Select the **combined_slip_test.xml** from the **shared database**



Tire Testrig

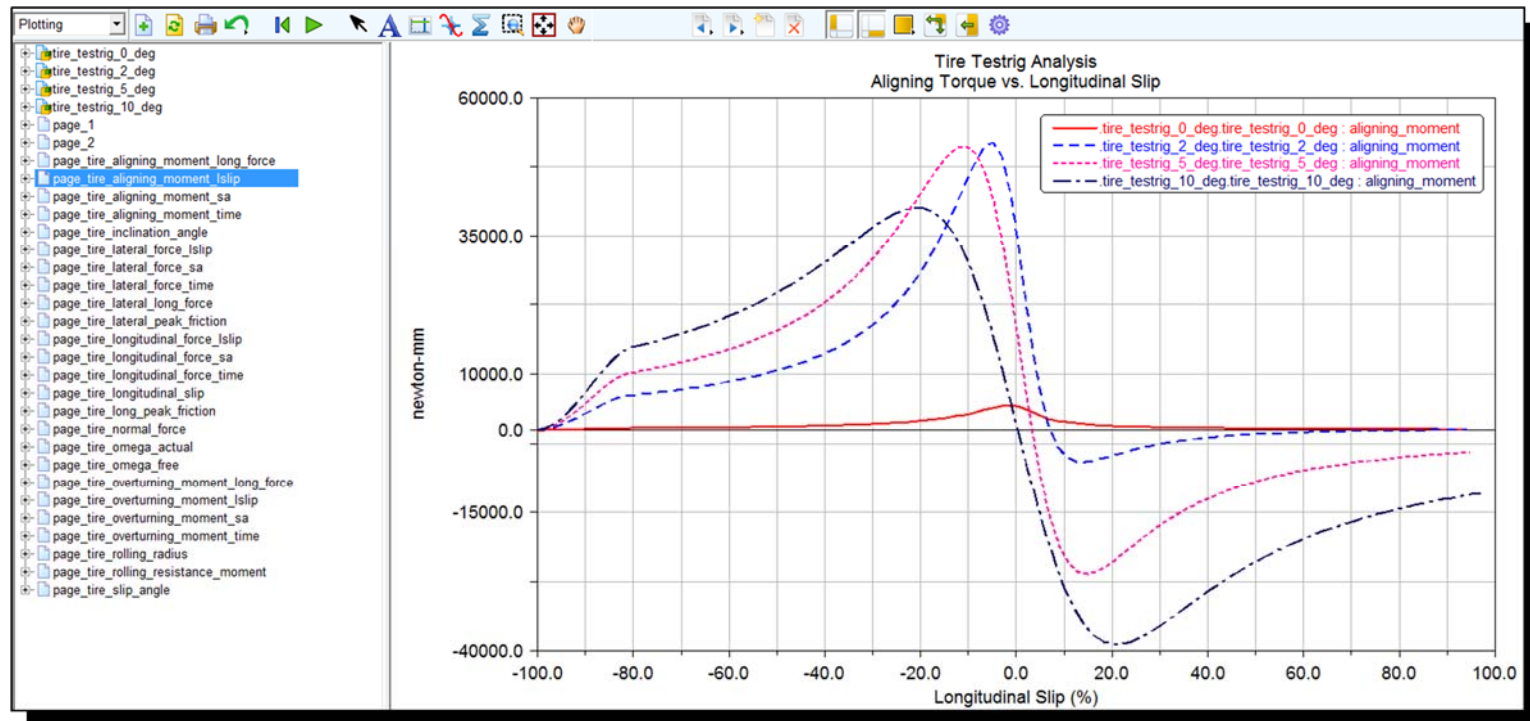
- **Run a Loadcase file with the tire testrig (Cont.):**
 4. The Tire Testrig screen shows the four analyses defined in the combined_slip_test.xml for the tire.



5. Click **Run It**. After submitting the job a dialog box pops up indicating the Tire Testrig plots are available.

Tire Testrig

- **Reviewing the results in Adams/Postprocessor:**
 - The Adams/Postprocessor is automatically displayed in plotting mode once the analysis is complete. In the tree view on the left highlight each page to display the plots of various tire curves.



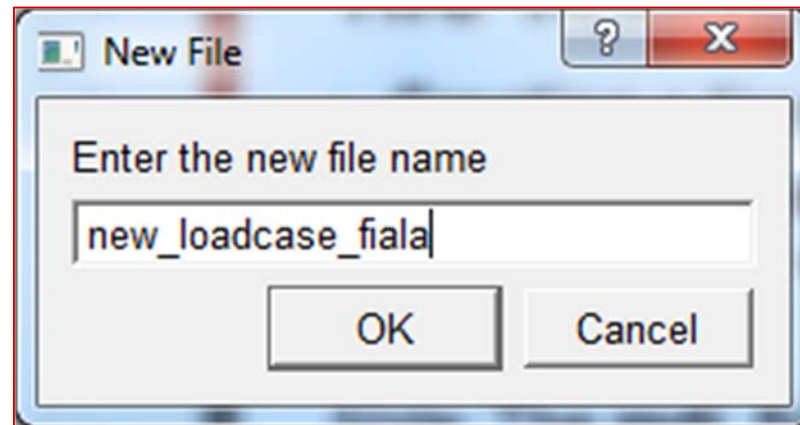
Tire Testrig

- **Creating a tire loadcase file:**

- You will create a new tire loadcase file. Here, you will compare two Fiala tire property files with different CSLIP parameter.

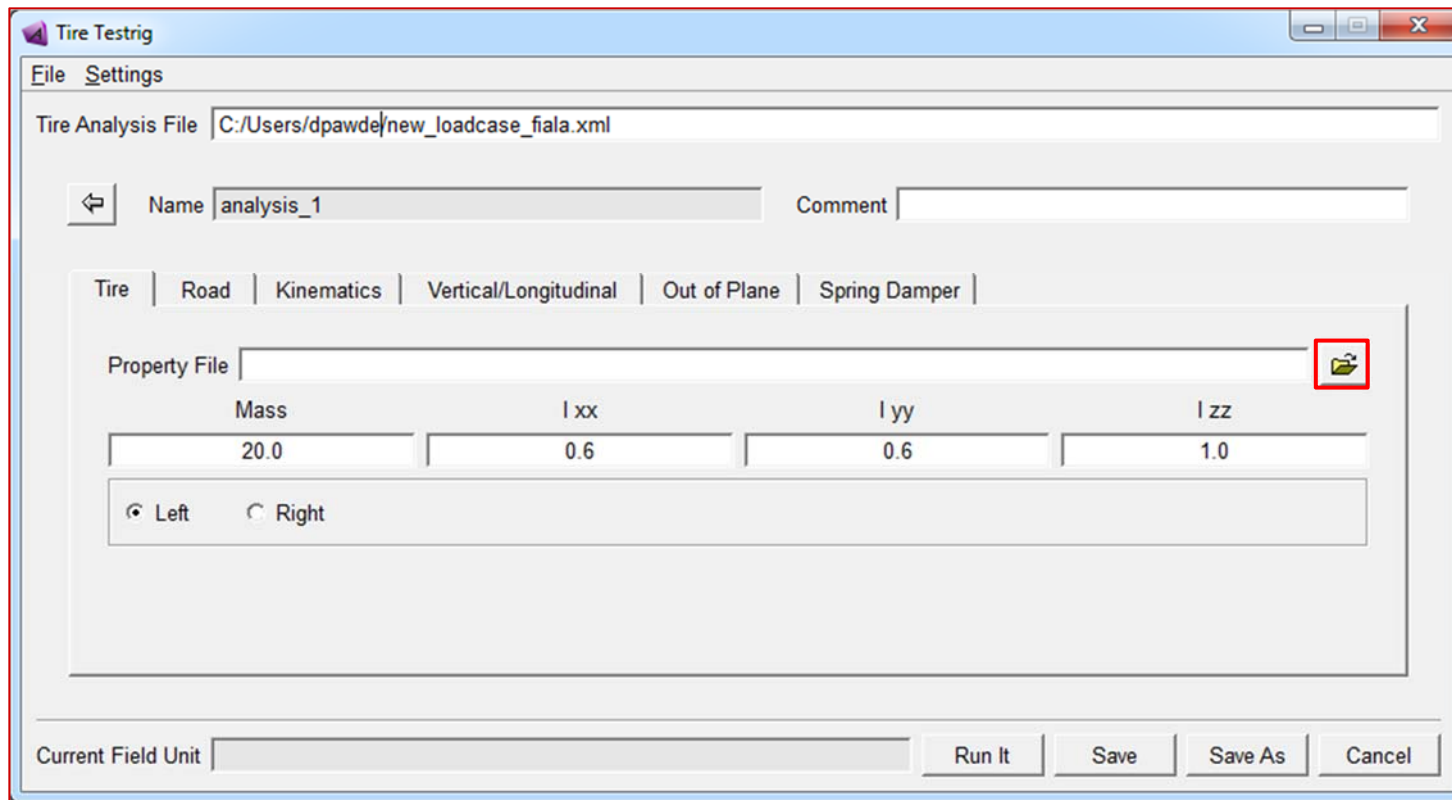
Note: The **mdi_fiala02.tir** required in this workshop is provided in your **acar_training_MD.cdb** database.

1. Open Adams/Car session
2. Click **Simulate → Component Analysis → Tire Testrig**
3. Click **File → New**. Enter name in the New File window as **new_loadcase_fiala**.
4. Click **OK**



Tire Testrig

- **Creating a tire loadcase file (Cont.):**
You will be prompted with following window.



The screenshot shows the 'Tire Testrig' software window with the 'Settings' tab selected. The 'Tire Analysis File' field contains the path 'C:/Users/dpawde/new_loadcase_fiala.xml'. Below this, there is a 'Name' field with 'analysis_1' and an empty 'Comment' field. A tabbed interface shows 'Tire' as the active tab, with other tabs including 'Road', 'Kinematics', 'Vertical/Longitudinal', 'Out of Plane', and 'Spring Damper'. Under the 'Tire' tab, there is a 'Property File' field with a folder icon button to its right. Below this, a table displays four properties: Mass (20.0), I xx (0.6), I yy (0.6), and I zz (1.0). At the bottom of the table, there are radio buttons for 'Left' (selected) and 'Right'. The bottom of the window features a 'Current Field Unit' field and four buttons: 'Run It', 'Save', 'Save As', and 'Cancel'.

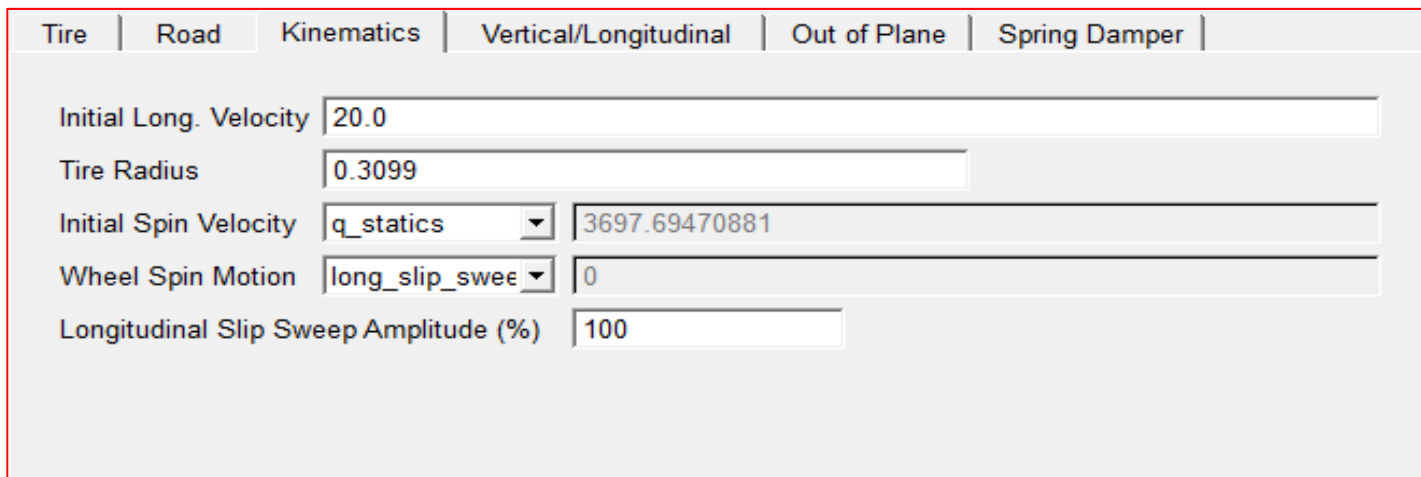
Mass	I xx	I yy	I zz
20.0	0.6	0.6	1.0

☒ Left ☐ Right

Tire Testrig

- **Creating a tire loadcase file (Cont.):**

5. Select “mdi_fiala01.tir” provided in acar_shared_database.
6. Click **Kinematics** tab.
7. In the **Tire Radius** field use the radius in the tire property file.
8. Change **Wheel Spin Motion** to have **Long_slip_sweep** and the **Amplitude** to be **100** as shown in the figure below.



The screenshot shows a software interface with a tabbed menu at the top: Tire, Road, Kinematics, Vertical/Longitudinal, Out of Plane, and Spring Damper. The Kinematics tab is selected. Below the tabs, there are five input fields with labels and values:

Field Label	Value
Initial Long. Velocity	20.0
Tire Radius	0.3099
Initial Spin Velocity	q_statics (dropdown) 3697.69470881
Wheel Spin Motion	long_slip_swee (dropdown) 0
Longitudinal Slip Sweep Amplitude (%)	100

9. Click the **Out of Plane** tab and set the **slip** and **inclination angle** values to be **0**

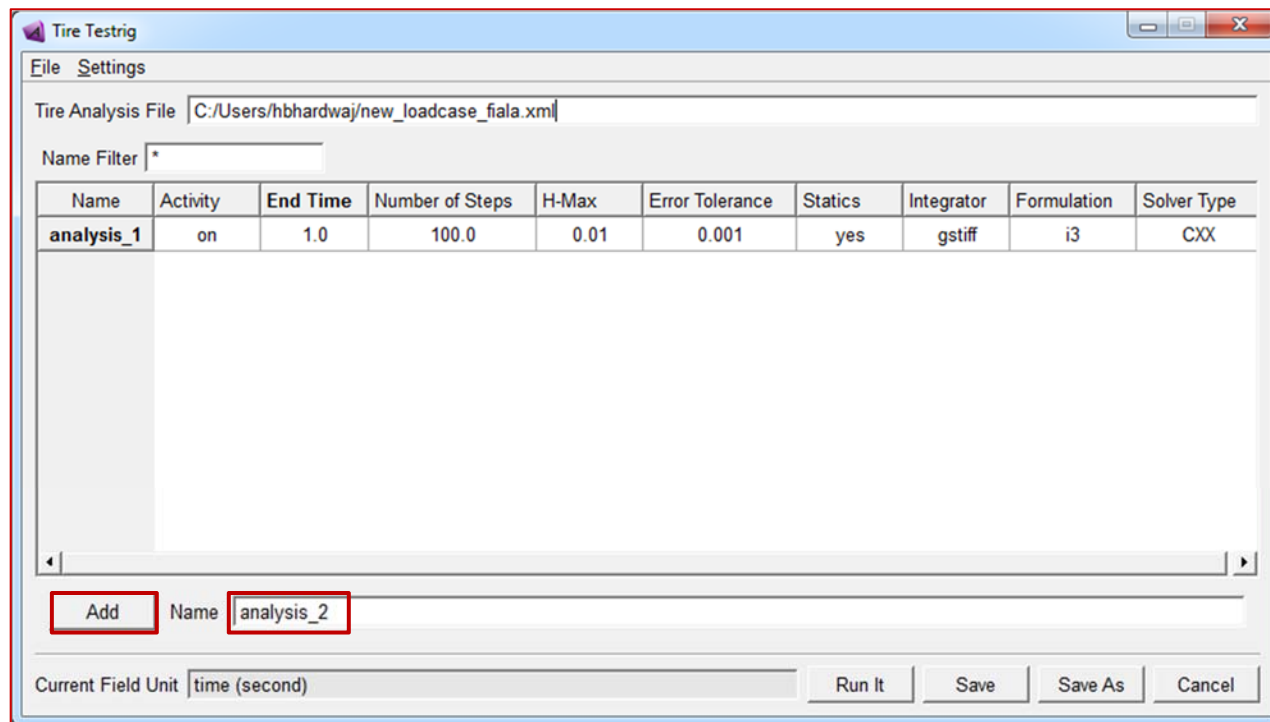
Tire Testrig

- **Creating a tire loadcase file (Cont.):**

10. Click on .

11. Enter name for the new analysis in **Name** field .

12. Click **Add** to add a second analysis **analysis_2**. This is where the user can modify solver parameters like End Time, Number of Steps, Error Tolerance, Integrator etc.



Tire Testrig

- **Creating a tire loadcase file (Cont.):**

13. Double click on **analysis_2** → Select **Tire tab** → Select tire property file as “**mdi_fiala02.tir**” from the database **acar_training_MD.cdb**.

14. This tire property has modified CSLIP parameter. Make sure the other inputs to the testrig match those for **analysis_1**.

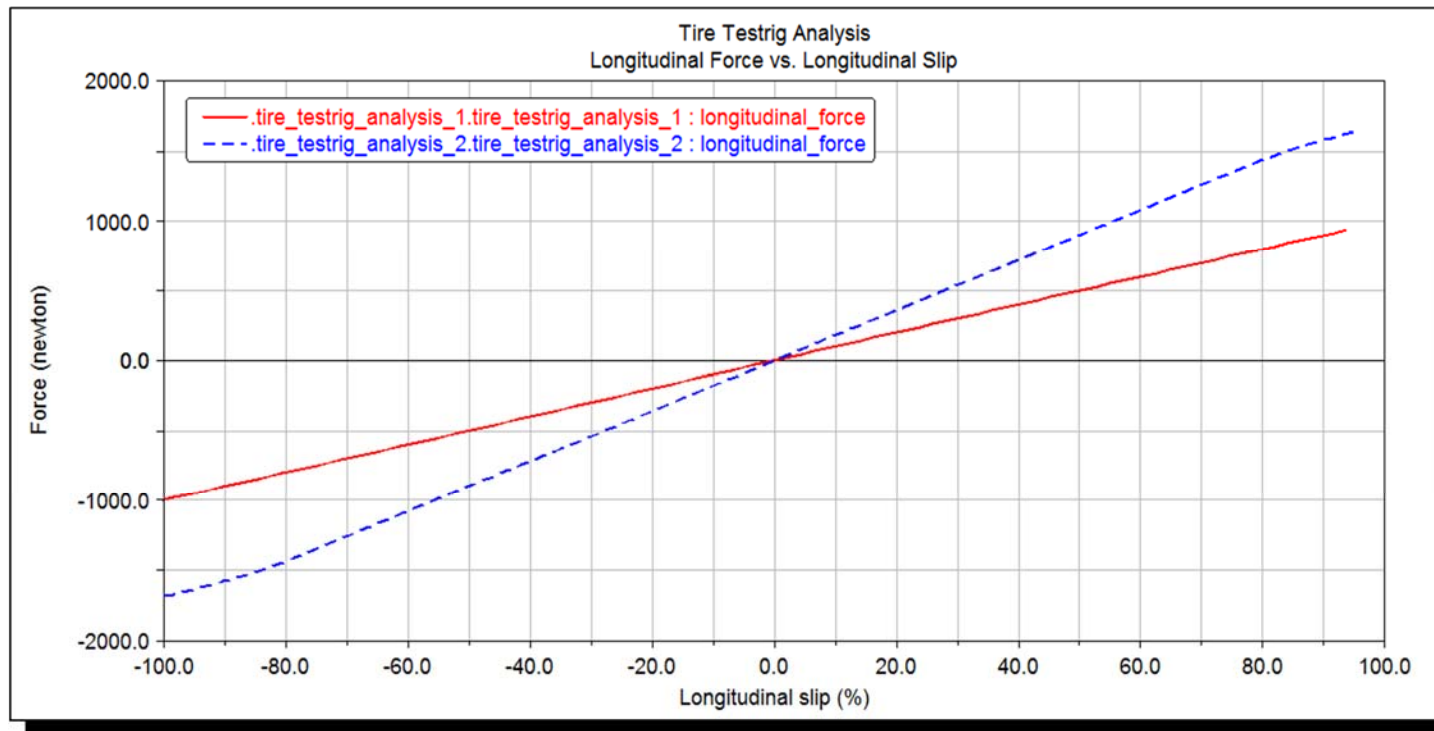
15. Click **Save As** to save this loadcase file in your **acar_training.cdb** directory.

16. Click **Run It**.

After submitting the job you get information window when the plots are ready. Click **OK**

Tire Testrig

- **Reviewing the results in Adams/Postprocessor:**
 - Adams/Postprocessor is launched when the plots are ready. Look at the Longitudinal Force vs. Slip plot. The red curve show the response when CSLIP=1000 and the blue one when CSLIP=1800.



Tire Testrig

- **Relevant Simcompanion Articles:**

- Additional and optional information on the brief overview of the parameters used as inputs by the testrig can be found in the Simcompanion article.
- **ATR-002: Using the Tire Testrig**
Simcompanion article #: KB8017771

<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8017771>

WORKSHOP 9

CREATING ROAD PROPERTY FILES USING ROAD BUILDER

Road Builder

- **Problem Statement:**

- In this workshop, you will learn how to create, visualize and edit the road data file, to create obstacles using road builder, and how to use that for various vehicle maneuvers.
- The workshop contains the following sections:
 - Study example Road data file
 - Creating new 3D Road data file
 - Creating 3D obstacles



This workshop takes one hour to complete.

Road Builder

- **Study Example Road Data File:**
 - Before you start creating a new road data file, it is recommended that you study an existing example file.
- **To open the Road Builder:**
 1. Open the assembly **MDI_Demo_Vehicle** from the **acar_shared** database.
 2. From the **Simulate** menu, point to **Full-Vehicle Analysis**, and then select **Road Builder**.
 3. Adams/Car displays the Road Builder, with default set **road_3d_sine_example.xml** road data file.

Road Builder

- **Road Builder Dialog Box:**

- Check with **Header, Global, Road Points, Obstacle Tabs** one by one

- **Header** – Display information related to file, allow you to set units, and lets you enter comments in the comment field.
 - **Global** – Used to set global setting such as Forward Direction, Search Algorithm, Road type, Road Vertical and Global parameters such as width, bank angle, and friction. (0,0,1) for road vertical means z axis as vertical which is inline with global reference frame.
 - **Soil Properties** – Used to simulate elastic/plastic tire-soil interactions. For more information see Simcompanion article #: KB8018590
<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8018590>
 - **Road Points** – Click on the road data point tab, check road data point coordinates, check the plots by clicking on show X-Z plot

Road Builder

- **Road Builder Dialog Box (Cont.):**
 - **Obstacle** - You should see the obstacle/s information in a tabular format. Click on obstacle name for more information. Note common parameters and obstacle specific parameters here.
 - **Road Generator** – New feature that automatically generates roads by describing section of the following types:
 - Linear
 - Curvature
 - Transition
 - User Function
 - User Points

Road Builder

- **Creating a new 3D Road data file:**
 1. From the **Simulate** menu, point to **Full-Vehicle Analysis**, and then select **Road Builder**.
 2. From **File** menu click on **New**. A newly created .xml file will be saved into your working directory.
 3. No need to change the units since we will be using default only.
 4. Click on **Global** tab, set the below given options
 - Forward Direction : Normal
 - Search Algorithm : Fast
 - Closed Road : No
 - Road vertical : 0.0, 0.0, 1.0
 5. The Global parameter related to road can either be set in Global tab or in the road points tab. We will set those in the Global tab as given on the next page.
- Note:** Global parameters take precedence over road points parameters.

Road Builder

- **Creating a new 3D Road data file (Cont.):**

- Global Parameters:

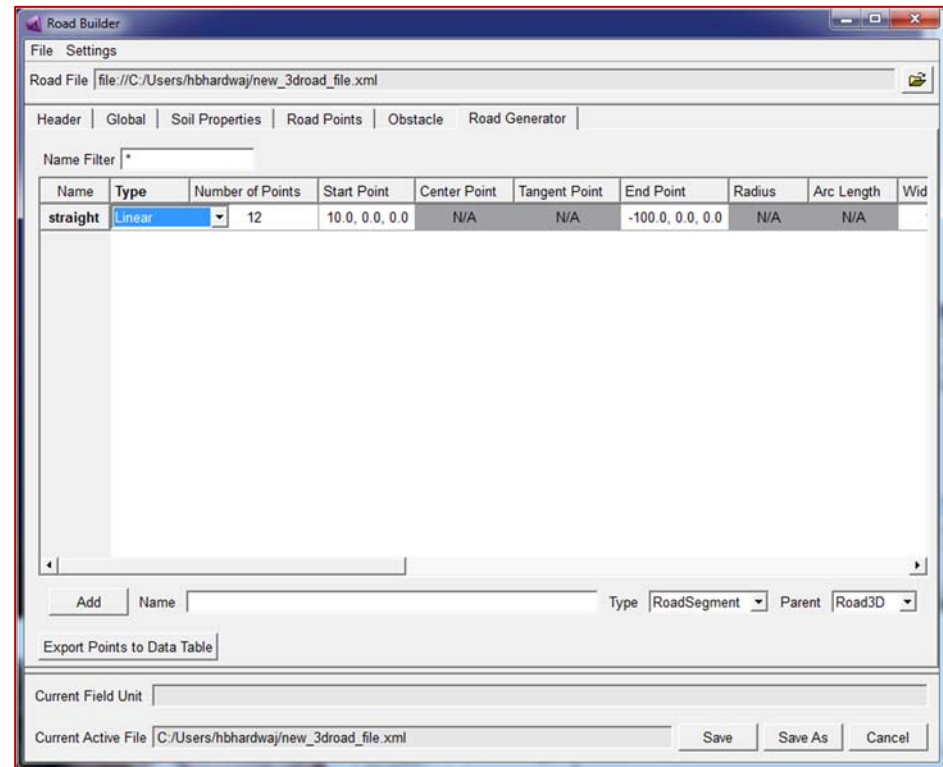
- Road Width : 12.0
 - Global Bank : 0.0
 - Road Friction Left : 0.9
 - Road Friction Right : 0.9

6. Click on **Road Generator** tab, Add a new segment named **straight** and enter the information as shown.

7. Click **Export Points to Data Table**.

8. Click the **Road Points** tab to view point table generated by Road Generator. Note that points can be entered and edited manually.

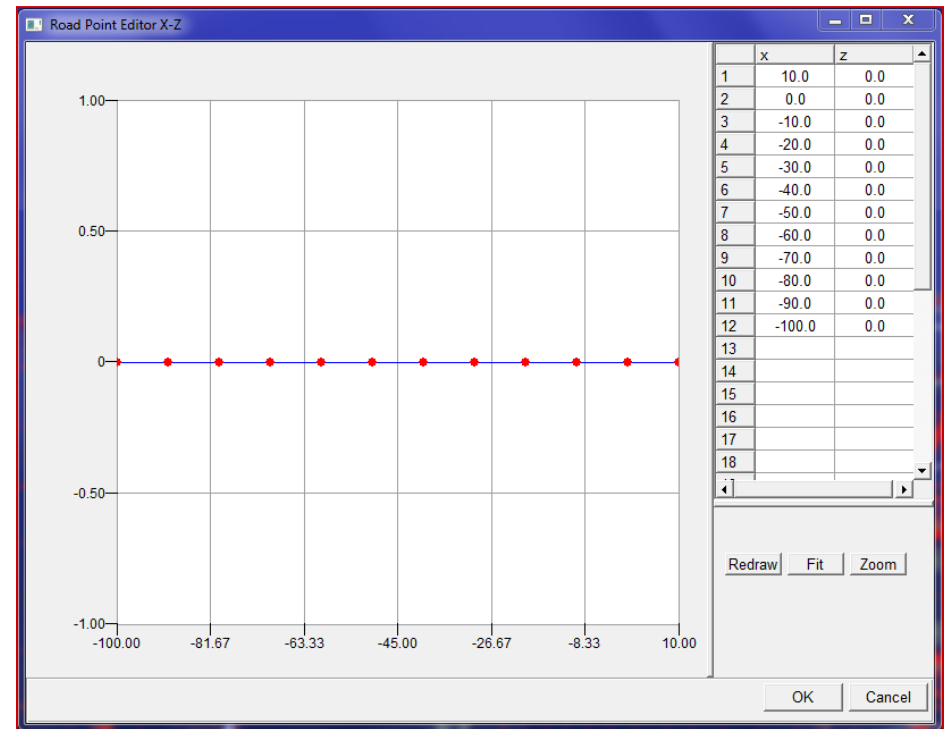
9. We have generated a straight segment of road 110 m long.



Road Builder

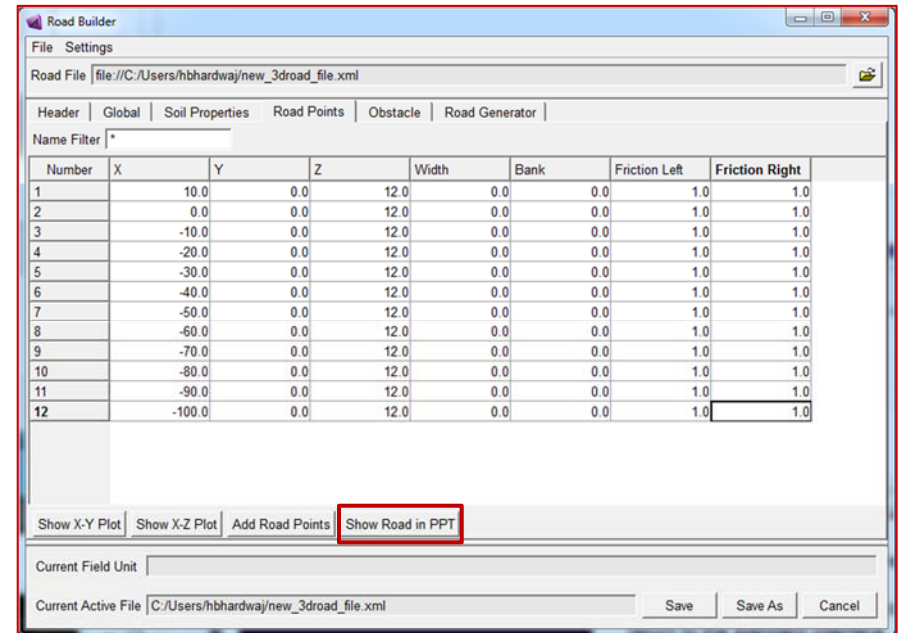
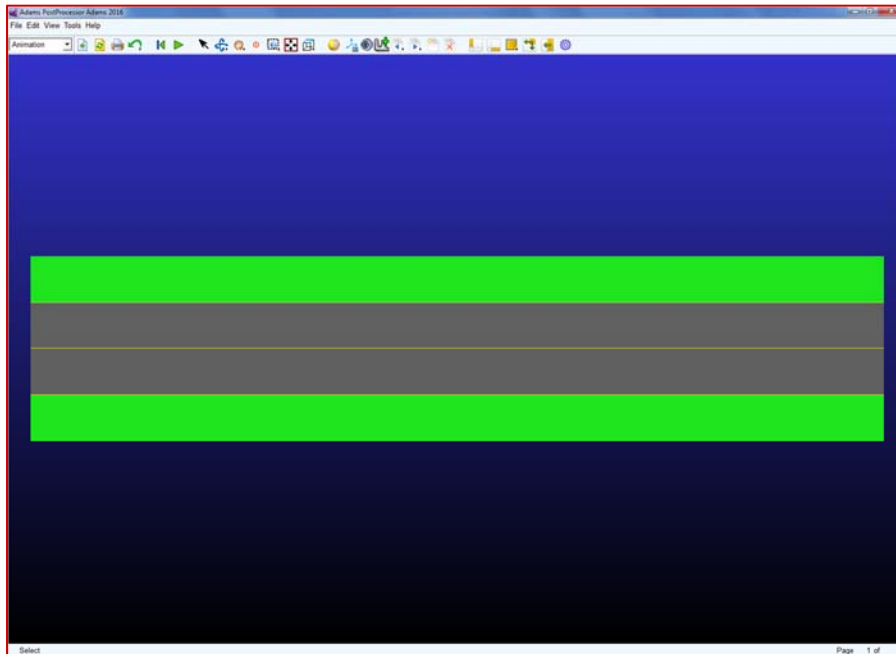
- **Creating a new 3D Road data file (Cont.):**

10. Click on **Show X-Z Plot** to see the spline. You will see a straight line as shown in the adjacent figure with view selected as Fit to entire plot size.
11. You can alter the shape of the spline by click and drag operation on the curve to get the shape you desire.
12. Click on **Save As** button to save this file into **acar_training** database. Select **acar_training** database from **Registered Databases** then select **roads.tbl** table. Save the file as **"3d_flat_road.xml"**.



Road Builder

- **3D Road generator with Adams:**
 1. You can use the Show Road in PPT functionality to check how your road going to look like in Adams/Car.
 2. Click on the **Show Road in PPT** tab at the bottom.

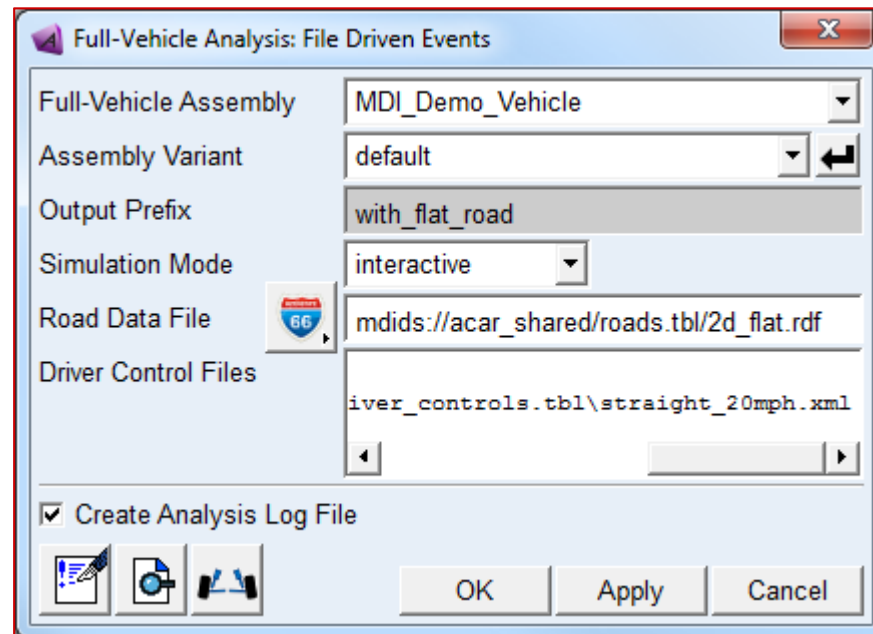


3. The road will be displayed on the Adams/Car GUI window.
4. Press **F8** or close this window to return to the standard interface.

Road Builder

- **Run a File Driven Analysis:**

1. Open **MDI_Demo_Vehicle**
2. Run a file driven event to simulate a steady state maneuver with certain velocity. **Simulate >> Full Vehicle Analysis >> File Driven Events**
3. Select just created road data file **3d_flat_road.xml**.
4. Select the **straight_20mph.xml** driver control file provided in the **acar_training_MD** database.

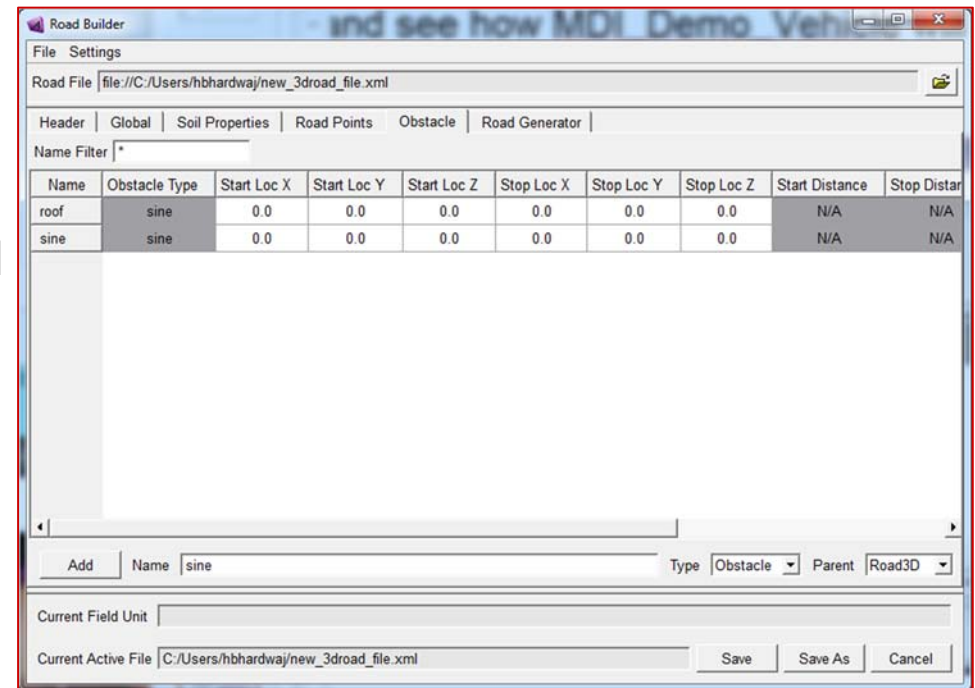


Road Builder

- **Creating 3D obstacles:**

- We are going to create a couple of obstacles of type roof and sine on the flat road we have just created and see how MDI_Demo_Vehicle will respond to these obstacles.

1. Click on **Obstacle** tab.
2. You will notice a blank list.
3. Enter the name of the obstacle as “roof” against the **name** field and click on the **Add** button to add the obstacle. It will appear as first entity in the list.
4. Add one more obstacle with name “sine”.



Road Builder

- **Creating 3D obstacles (Cont.):**

5. Double-click on “**roof**”
6. Enter the parameters as shown in the image below.
7. Click on arrow button to go back to the list.

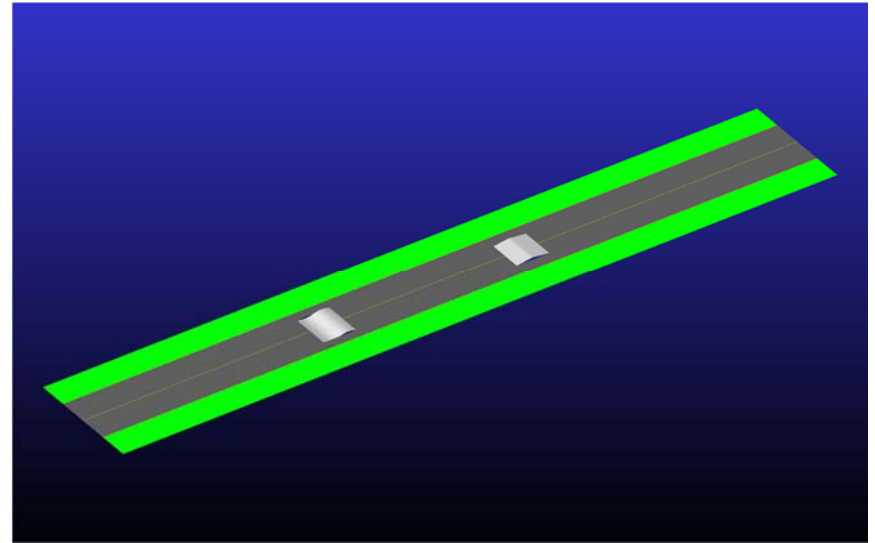
The screenshot shows the 'Road Builder' application window with the 'Obstacle' tab selected. The 'Name' field is set to 'roof'. Under 'Common Parameters', 'Obstacle Type' is 'roof', 'Friction' is 0.9, 'Length' is 5, and 'Width' is 7. The 'Coordinate System' is set to 'Local'. The 'Start Location' is (-30, 0.0, 0.0) and the 'Stop Location' is (-40, 0.0, 0.0). Under 'Obstacle Specific Parameters', the 'Height' is 0.3. The 'Current Field Unit' is 'length (meter)' and the 'Current Active File' is 'training.cdb/roads.tbl/3d_flat_road.xml'.

8. Double-click on “**sine**”
9. Enter the parameters as shown in the image below.
10. Click on the **arrow button** to go back to the list.

The screenshot shows the 'Road Builder' application window with the 'Obstacle' tab selected. The 'Name' field is set to 'sine'. Under 'Common Parameters', 'Obstacle Type' is 'sine', 'Friction' is 0.9, 'Length' is 5, and 'Width' is 8. The 'Coordinate System' is set to 'Local'. The 'Start Location' is (-60, 0.0, 0.0) and the 'Stop Location' is (-70, 0.0, 0.0). Under 'Obstacle Specific Parameters', the 'Amplitude' is 0.2, 'Wavelength' is 0.5, and 'Wavelength Offset' is 0.0. The 'Current Field Unit' is 'length (meter)' and the 'Current Active File' is 'training.cdb/roads.tbl/3d_flat_road.xml'.

Road Builder

- **Creating 3D obstacles (Cont.):**
 - Click on **Road Points** tab.
 - Click on **Show Road in PPT**.
 - You will see 3D road with a couple of obstacles in the Adams/Car GUI.
 - Save the file as
“**3d_road_with_obstacles.xml**”
 - You can modify the obstacles and see the changes through road generating capability.

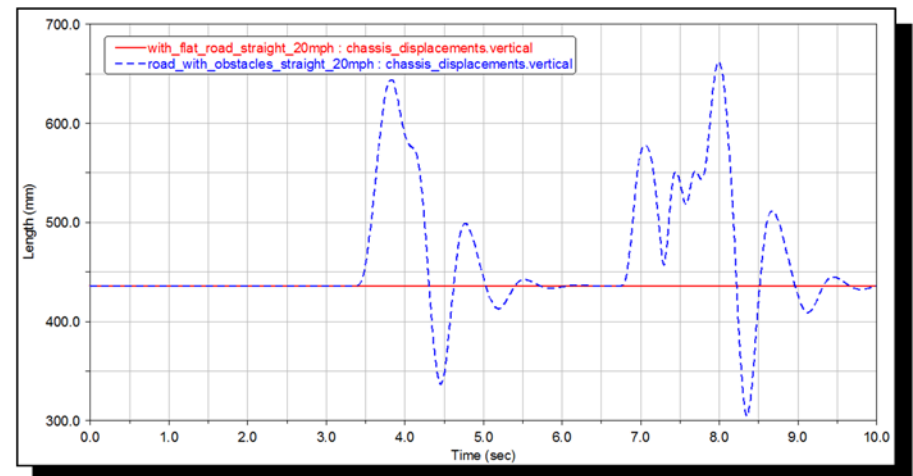
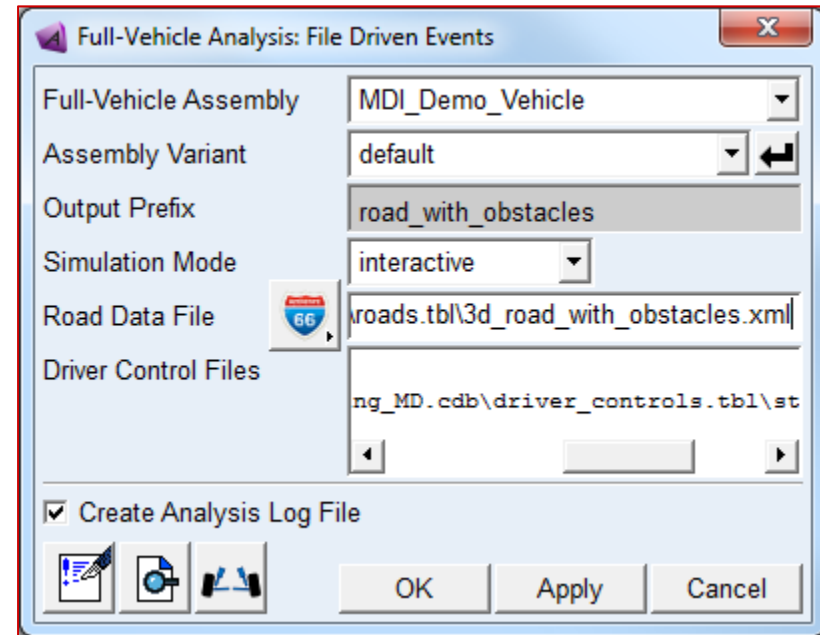


Note: The road and obstacles are created as shell objects attached to the ground.

Road Builder

- **Run a File Driven Analysis:**

- Run file driven event to simulate steady state maneuver with certain velocity. **Simulate >> Full Vehicle Analysis >> File Driven Events**
- Select just created road data xml file **3d_road_with_obstacles.xml** for road data.
- Select the **straight_20mph.xml** driver control file.
- Compare the vertical displacement of the CG of chassis to see the effect of obstacle. You can also plot wheel forces to see the effect of obstacles.



Road Builder

- **For more in-depth information and tutorials see Simcompanion Articles:**

- **ACR-007: Road Builder Enhancements**

- Simcompanion article #: KB8018223

- <https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8018223>

- **Deformable Road for Soft Soil Tire Modeling**

- Simcompanion article #: KB8018590

- <https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8018590>



WORKSHOP 10

FLEX TUTORIAL

Flex Tutorial

- **Problem Statement:**

- In this workshop, you swap a rigid lower control arm with a flexible control arm in a suspension subsystem.

- **First, you need to change the memory for A/View and A/Solver to be huge if you intend to use the Fortran Solver.**

- Windows: Start Menu → Programs → MSC.Software → Adams 2017.2 → Settings & license – ASolver and AView – Preferences – **MemSize = Huge**
 - Unix: Right Click Adams Toolbar – Start Registry Editor - ASolver and AView – Preferences – **MemSize = Huge**

Note: You need to start a new Adams/Car session after this change. Also, be sure to make the changes in both the Adams Solver and Adams View preferences folders.



This workshop takes about one and a half hours to complete.

Flex Tutorial

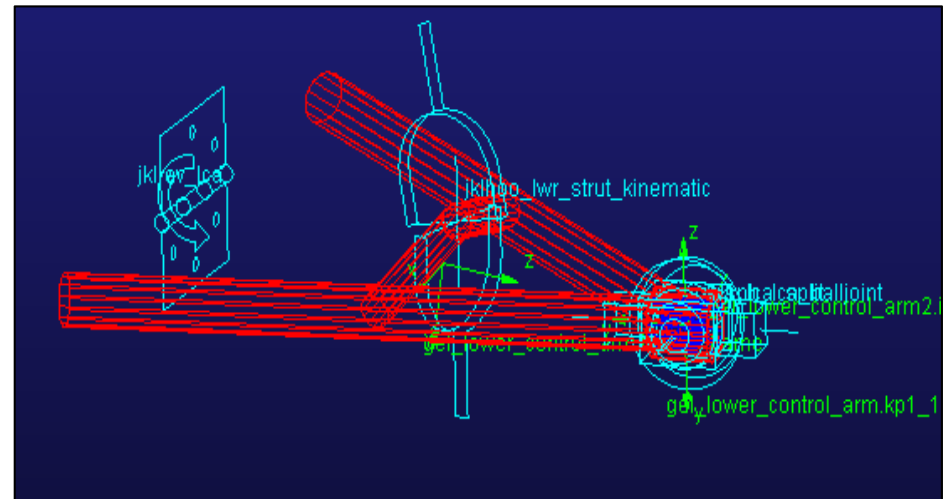
- Opening a subsystem

To open a subsystem:

1. From the **File** menu, select **Open**.
2. Right-click the Subsystem **Name** text box, point to **Search**, and then select **<acar_shared>\subsystems.tbl**.
3. Double-click **TR_Front_suspension.sub**.
4. Select **OK**.



Flex Tutorial

- **To create the flexible lower control arm:**
 1. From the **Adjust** menu, point to **General Parts**, point to **Rigid-to-Flex**.
Or, from the Model Browser, go to **Parts>General Parts**, right click the part you want to make flexible, select **Make Flexible**. Select **Import MNF** when prompted.
 2. In the Current Part Name text box, select **gel_lower_control_arm**.
 3. In the **Modal Neutral File** text boxes, point to the files **LCA_left_shl.mnf** , located in the **<acar_training_MD.cdb>** database.
 - In the main window, Adams/Car displays the flexible bodies on top of the rigid bodies. You can see bodies in this configuration when the model is displayed in wireframe mode.
- Tip:** Press **Shift + S** with the graphics area selected to toggle between wireframe and shaded mode.



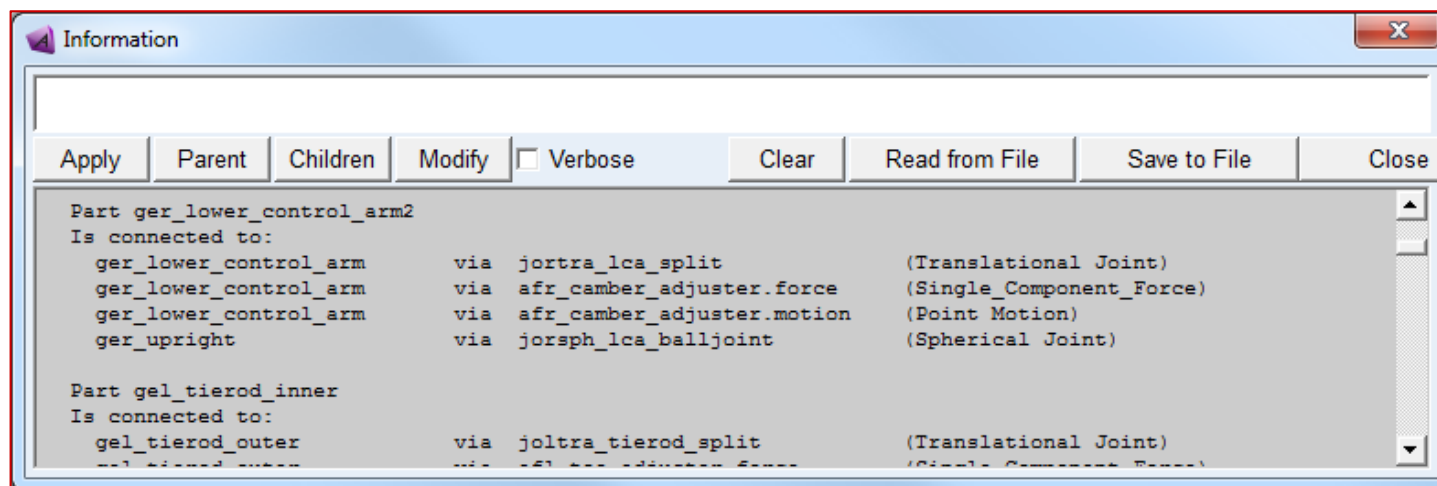
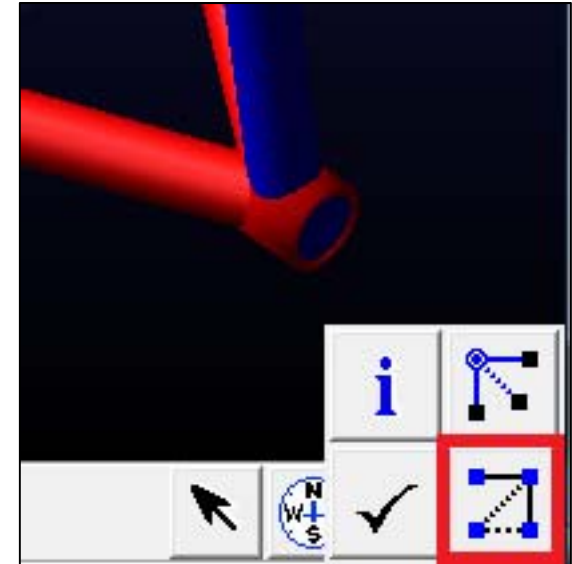
Flex Tutorial

- **Identify the attachments:**

1. On the **Status** bar of the main menu, right-click the **Information** tool stack, , and then select the **Model Topology** by **Parts** tool .

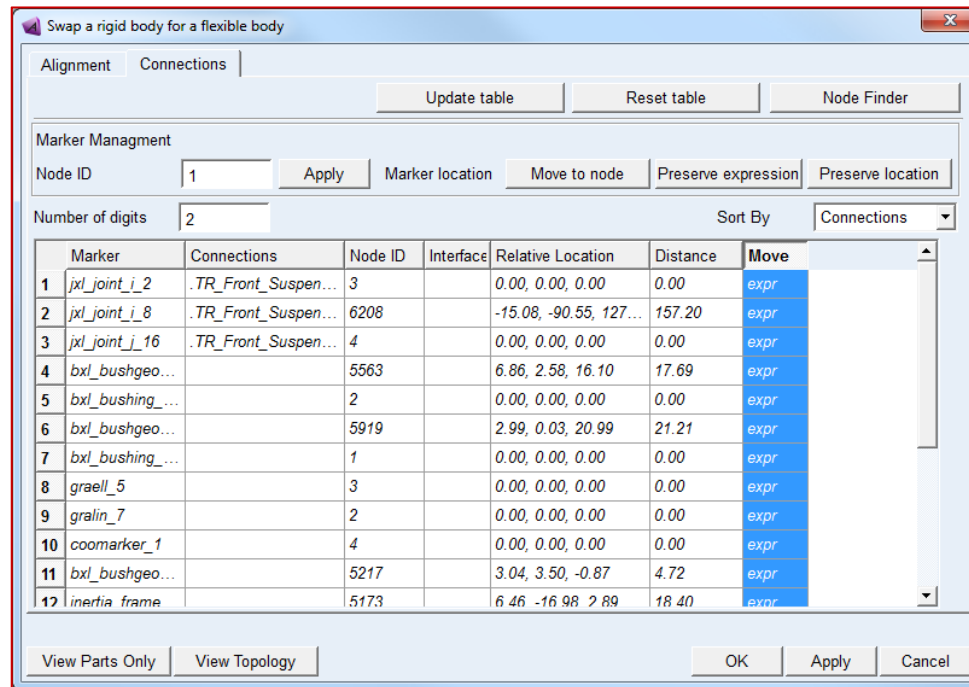
The Information window displays the topology map.

2. Scroll to the top of the **Information** window and find **gel_lower_control_arm**, the associated list of attachments, and their corresponding parts.



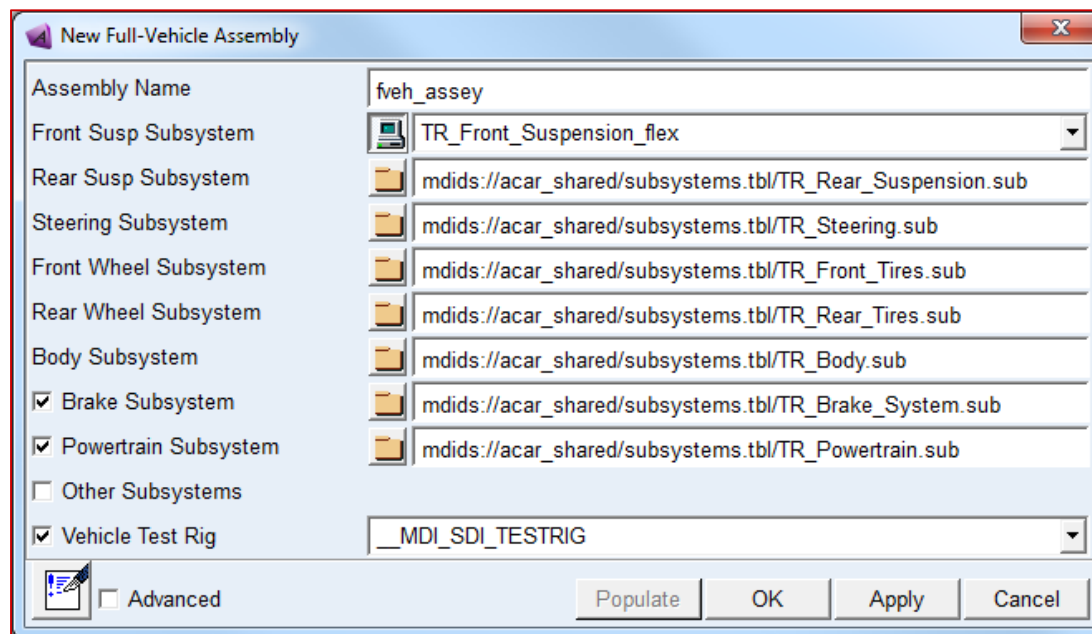
Flex Tutorial

- Associate connections to flexible body using Preserve Expression
 1. Select the **Connections** tab in the **Rigid/Flex Swap** window
 2. Select the **Move** tab to highlight all connections.
 3. Next select **Preserve expression**.
 4. Click **Apply** and repeat the process for the right control arm.



Flex Tutorial

- To run an analysis with your suspension with flexible control arm:
 1. Save the subsystem as **TR_Front_Suspension_flex** to the **acar_training_MD** database.
 2. Create the assembly
 - From the **File** menu, point to **New**, and then select **Full-Vehicle Assembly**.
 - Fill in the dialog box as shown next, and then select **OK**.
 - Click on the folder icon in Front Susp Subsystem field, to use the present subsystem in memory for the full vehicle model.



Flex Tutorial

- **Changing Modal Content**

- By default, when you integrate an MNF into an Adams/Car template, Adams/Flex enables all the modes that were defined during the FEM modeling except the probable rigid body modes. It is important to **have the right modal content in flexible bodies**, because an MNF has more modes than are needed to simulate a particular response.
- To increase the efficiency of the simulations, you could **disable any modes that do not contribute to the motion that your flexible part** will undergo during the simulation. Be careful when disabling modes, because **a disabled mode corresponds to a constraint to the part**. Changing the modal content of a flexible body corresponds to a flexible body setup. In addition, sometimes, it is not possible to know the contribution of one particular mode to the overall deformation of the flexible body.


Flex Tutorial

- You can manually toggle modes on or off. This gives you greater flexibility and helps you avoid potential problems. You can enable and disable modes in several ways:
 - Individually, based on their mode number.
 - As a group, based on their mode number or frequency.
 - Through a table editor. The table editor also lets you define displacement and velocity initial conditions associated with every mode.
 - Based on their strain energy contribution, but you can only do this after a successful analysis. For more information on this technique, see the Adams/Flex online help.

Flex Tutorial

- **To disable individual modes:**

- Some of the modes of the flexible lower control arms do not contribute effectively to the dynamic behavior of the entire system. We recommend that you disable them to reduce the computational effort and to improve the efficiency of the simulation.
- You must disable these modes for both the left and the right side because a left and a right MNF defines the flexible lower control arms.

1. Zoom in on the front suspension.
2. Right click the **gel_lower_control_arm_flex**, point to **Modify**, and then select modify icon .

The Flexible Body Modify dialog box appears.

Flex Tutorial

3. Select **Modal ICs**.

The Modify Modal ICs dialog box appears.

4. Hold down the **Shift** key, select modes **28** and **29**, and then select **Disable Highlighted Modes**.
5. Close the Modify Modal ICs dialog box.
6. Repeat Steps 2 through 6, for the right lower control arm, **ger_lower_control_arm_flex**. (You could also just right-click the **Flexible Body** text box, point to **Flexible Body**, point to **Pick**, and from the screen select **ger_lower_control_arm_flex**.)
7. Close the Flexible Body Modify dialog box.

Flex Tutorial


- **Performing a Full-Vehicle Analysis**

- You are now ready to perform the full-vehicle analysis. After you perform the analysis, you can change the inertia modeling of the flexible body to compare the effect of the modal flexibility on the dynamics of the vehicle.

- **To perform the full-vehicle analysis:**

1. From the **Simulate** menu, point to **Full-Vehicle Analysis**, point to **Open-Loop Steering Events**, and then select **Step Steer**.
2. Set up the analysis as follows:
 - **Full-Vehicle Assembly:** `fveh_assy`
 - **Output Prefix:** `tst`
 - **End Time:** `4`

Flex Tutorial

- Number of Steps: 400
 - Initial Velocity: 70 (take the default of km/hr)
 - Gear Position: 6 (Use  to select the appropriate gear position)
 - Final Steer Value: -45
 - Step Start Time: 1
 - Duration of Step: 1
 - Steering Input: Angle
3. Keep the defaults for **Cruise Control (off)** and **Quasi-Static Straight Line Setup (on)**.
 4. Select **Apply**, so the dialog box stays open for the analysis you will run in the next section.

The SDI test rig applies to the assembly the inputs you specified and performs a dynamic analysis.

Flex Tutorial

- **To change the inertia modeling:**
 1. Right click the **gel_lower_control_arm_flex**, point to **Modify**, and then select modify icon to bring up the **Flexible Body Modify** dialog box.
 2. Set **Inertia modeling** to **Rigid body**.
 3. Select **OK**.
 4. Close the Modify Flexible Body dialog box.
 5. Repeat Steps 1 through 4, for the right lower control arm, **ger_lower_control_arm_flex**.
 6. Submit another step steer analysis using the same inputs as before, but changing the output prefix to **tst_rigid**.
Adams/Car analyzes the assembly.

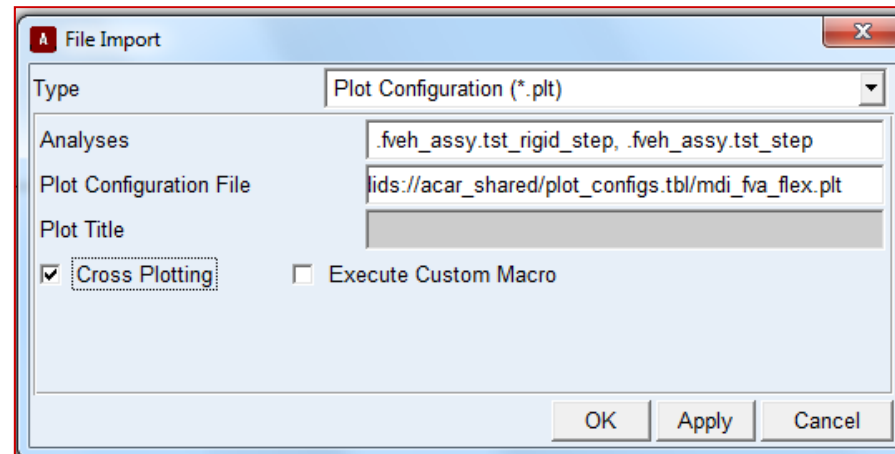
Flex Tutorial

- **Plotting Analysis Results**

- In this section, you create a set of plots that show the behavior of your vehicle assembly and then review how the modal flexibility of the lower control arm affects the overall dynamics of the vehicle.

- **To plot the results:**

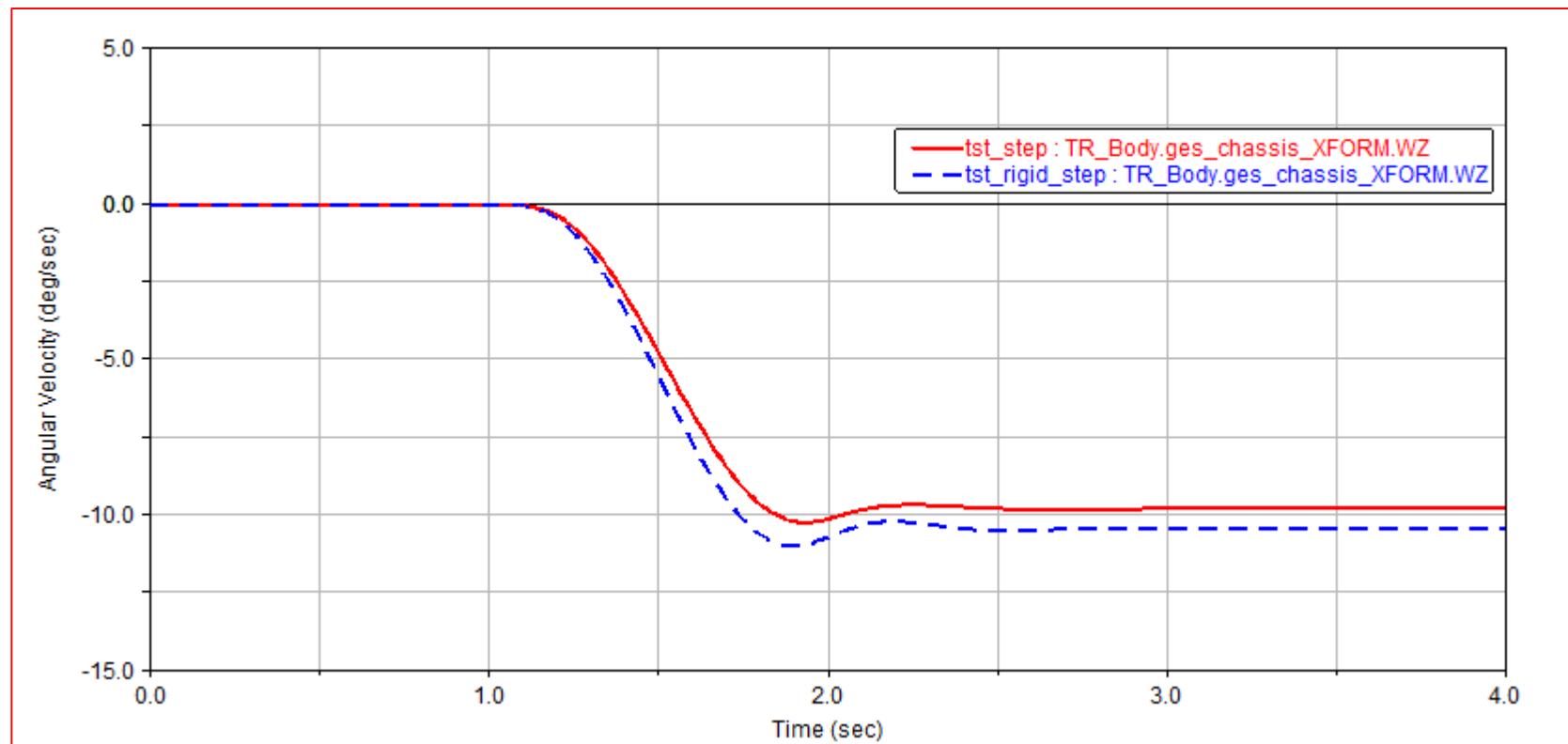
1. Launch Adams/PostProcessor.
2. From the **Plot** menu, select **Create Plots**.
3. Fill in the dialog box as shown next, and then select **OK**.



Flex Tutorial

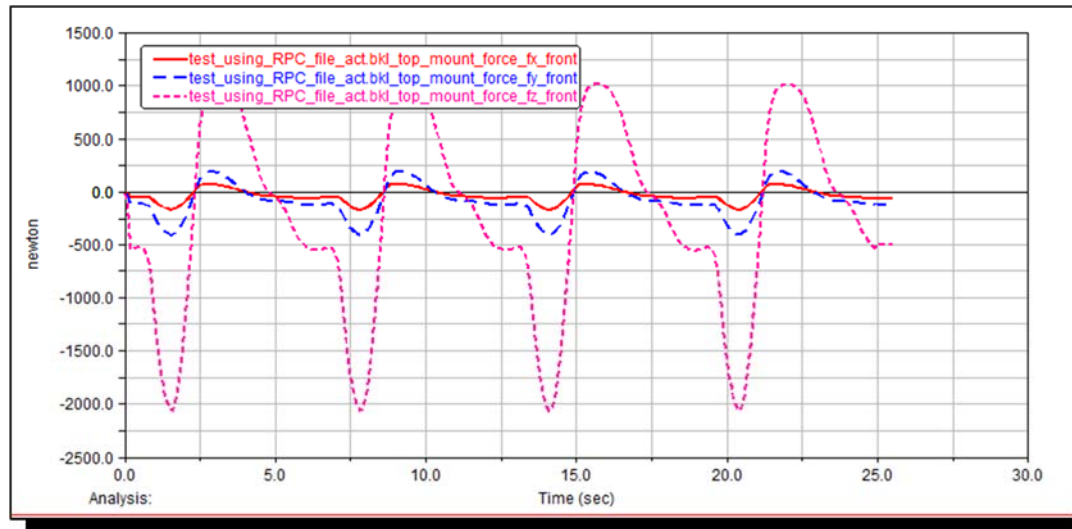
- Figure shows an example. The flexible lower control arms cause the differences between the curves for the Yaw Rate of the vehicle.

4. View the plots shown next and then return to Adams/Car.



WORKSHOP 11

GENERAL ACTUATION ANALYSIS



General Actuation Analysis

- **Problem statement:**

- In this workshop you will:

- Perform actuation analysis on an assembly, review animations and view plots in Adams/Postprocessor.
 - Perform actuation analysis using RPC3 files and view results.



This workshop takes one hour to complete.

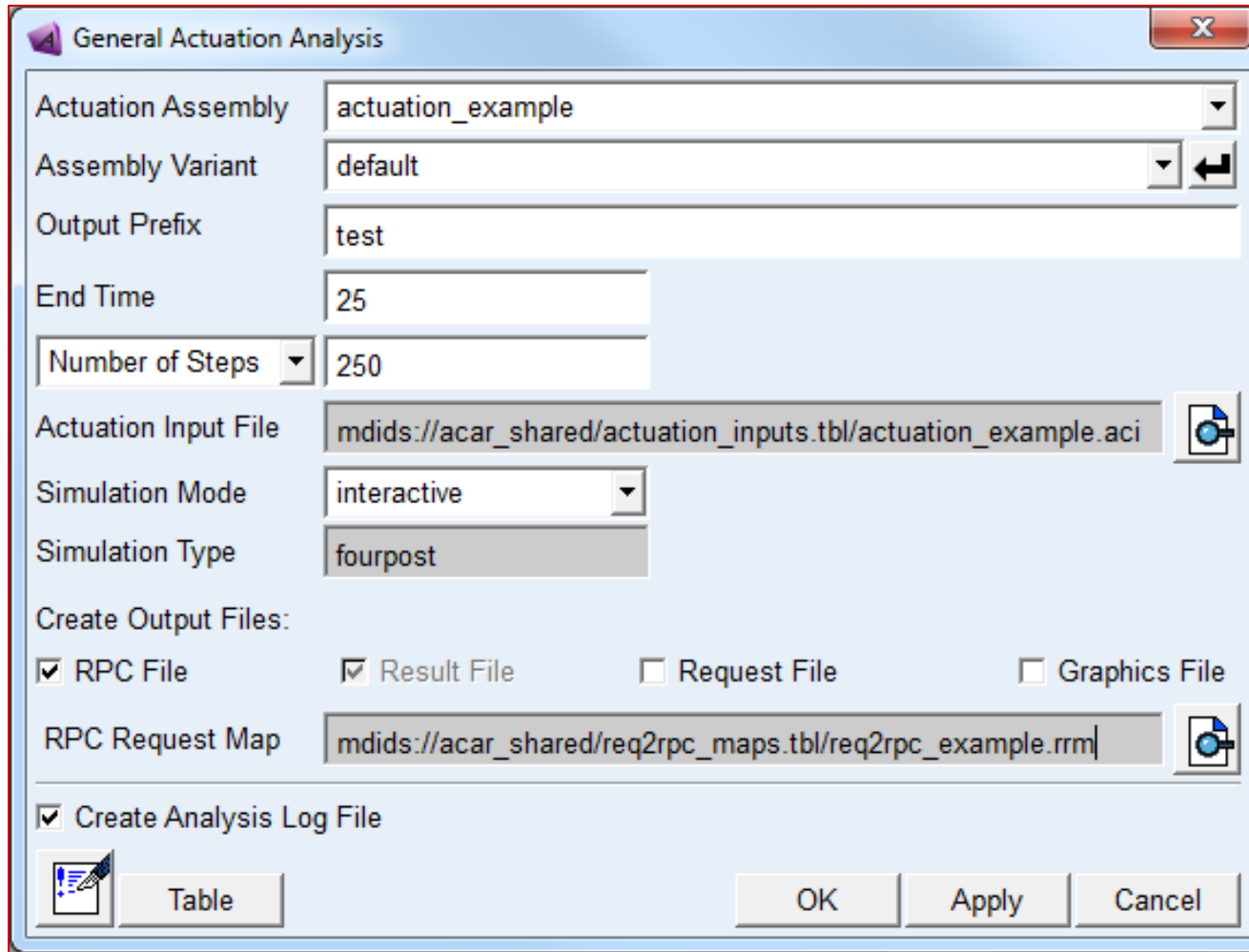
Perform actuation analysis on assemblies

- **To run the General Actuation Analysis:**
 1. Start Adams/Car in Standard Interface.
 2. Open the assembly **actuation_example.asy** from **acar_shared** database.
 3. From the **Simulate** menu, select **General Actuation Analysis** → **Submit Analysis**.
 4. Enter the parameter in the dialog box as shown on next slide.
 5. Select the Actuation Input file **actuation_example.aci** from **acar_shared** database.
 6. Select the RPC request map file **req2rpc_example.rrm** from **acar_shared** database.

Note: The actuator setup is done through the 'RPC request map file' and the 'Actuation input file'. Sample files have been provided with the installation and may be used as an example.

Perform actuation analysis on assemblies

7. Make sure the checkbox for **RPC File** and **Result File** are selected.
8. Select **OK**.



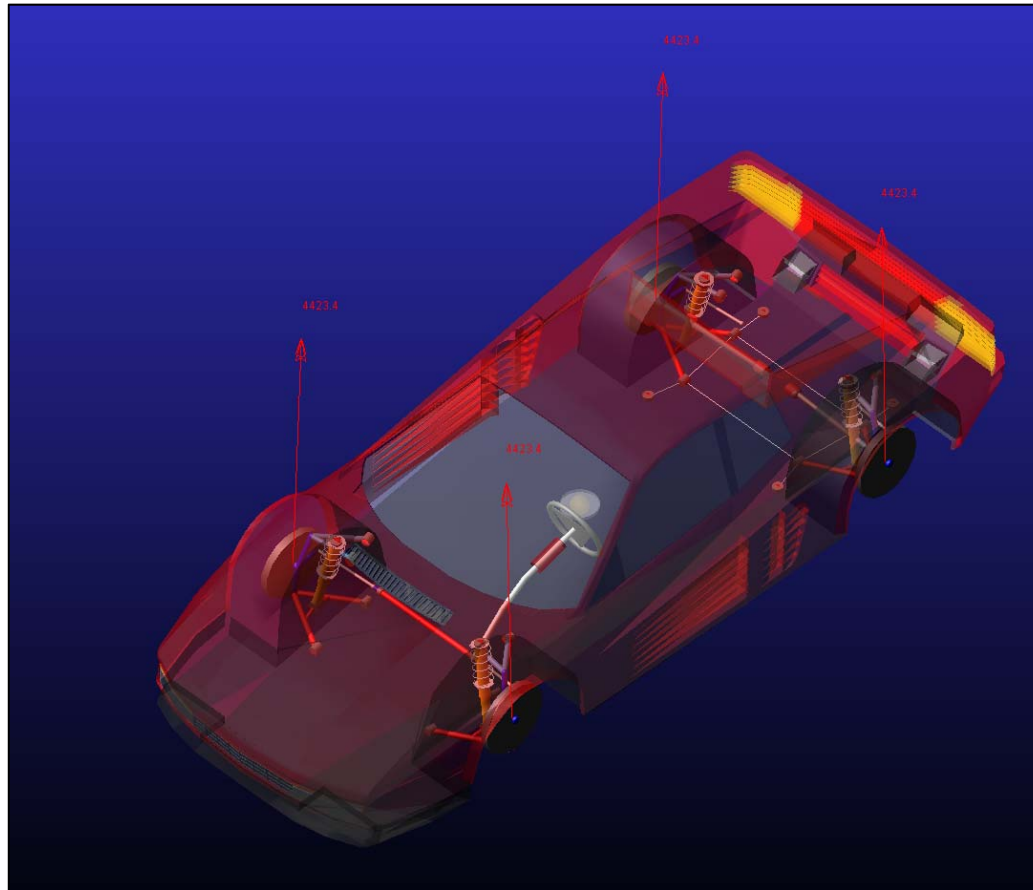
The image shows a software dialog box titled "General Actuation Analysis". It contains several input fields and checkboxes for configuring an actuation analysis. The fields are: "Actuation Assembly" (dropdown menu with "actuation_example" selected), "Assembly Variant" (dropdown menu with "default" selected), "Output Prefix" (text field with "test"), "End Time" (text field with "25"), "Number of Steps" (dropdown menu with "250" selected), "Actuation Input File" (text field with "mdids://acar_shared/actuation_inputs.tbl/actuation_example.aci" and a file selection icon), "Simulation Mode" (dropdown menu with "interactive" selected), and "Simulation Type" (text field with "fourpost"). Below these fields is a section titled "Create Output Files:" containing four checkboxes: "RPC File" (checked), "Result File" (checked), "Request File" (unchecked), and "Graphics File" (unchecked). At the bottom of this section is the "RPC Request Map" text field with "mdids://acar_shared/req2rpc_maps.tbl/req2rpc_example.rrm" and a file selection icon. At the very bottom of the dialog is a checkbox labeled "Create Analysis Log File" which is checked. There are three buttons at the bottom right: "Table" (with a file icon), "OK", "Apply", and "Cancel".

Actuation Assembly	actuation_example
Assembly Variant	default
Output Prefix	test
End Time	25
Number of Steps	250
Actuation Input File	mdids://acar_shared/actuation_inputs.tbl/actuation_example.aci
Simulation Mode	interactive
Simulation Type	fourpost
Create Output Files:	
<input checked="" type="checkbox"/> RPC File	<input checked="" type="checkbox"/> Result File
<input type="checkbox"/> Request File	<input type="checkbox"/> Graphics File
RPC Request Map	mdids://acar_shared/req2rpc_maps.tbl/req2rpc_example.rrm
<input checked="" type="checkbox"/> Create Analysis Log File	

Table OK Apply Cancel

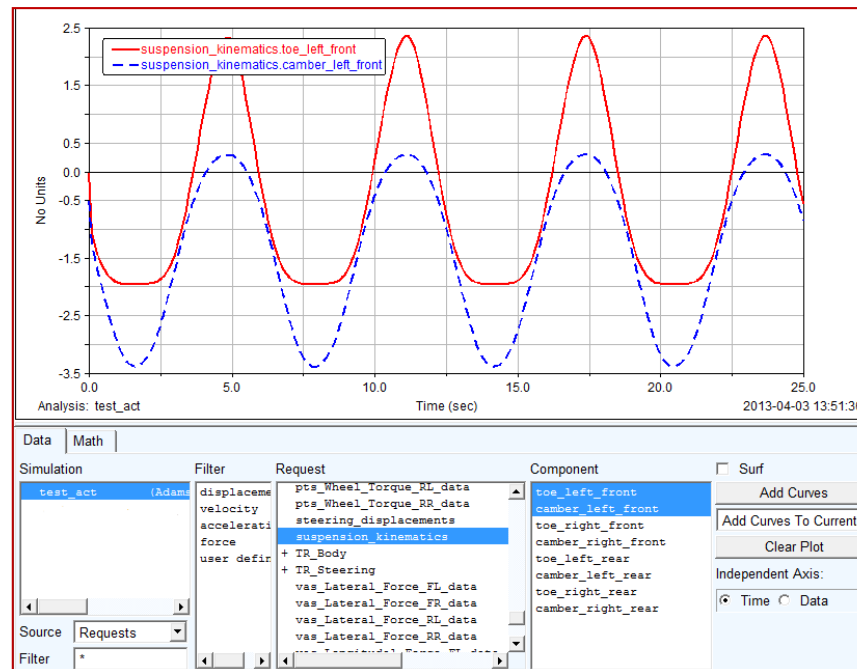
Review Animation

- To view the animation, select **Review** → **Animation Control**, animate the model and observe the actuation of wheel and suspension.



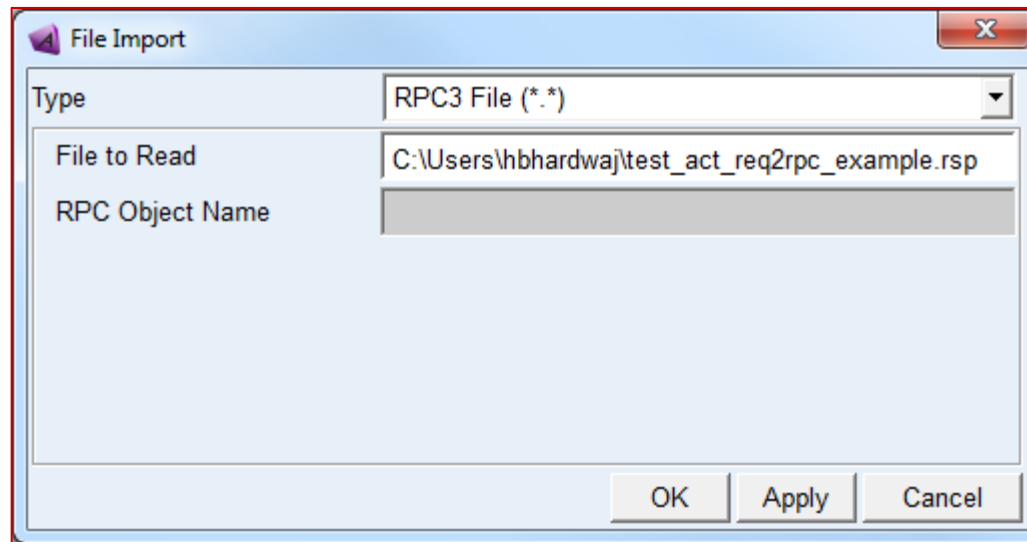
Plotting results in Adams Postprocessor

1. To view the result of the analysis switch to Post-processing window (press **F8**)
2. Select **suspension_kinematic** under **Request** and **toe_left_front** and **camber_left_front** with **control key** under **Component** menu.
3. Select **Add Curves**.



Plotting results in Adams Postprocessor

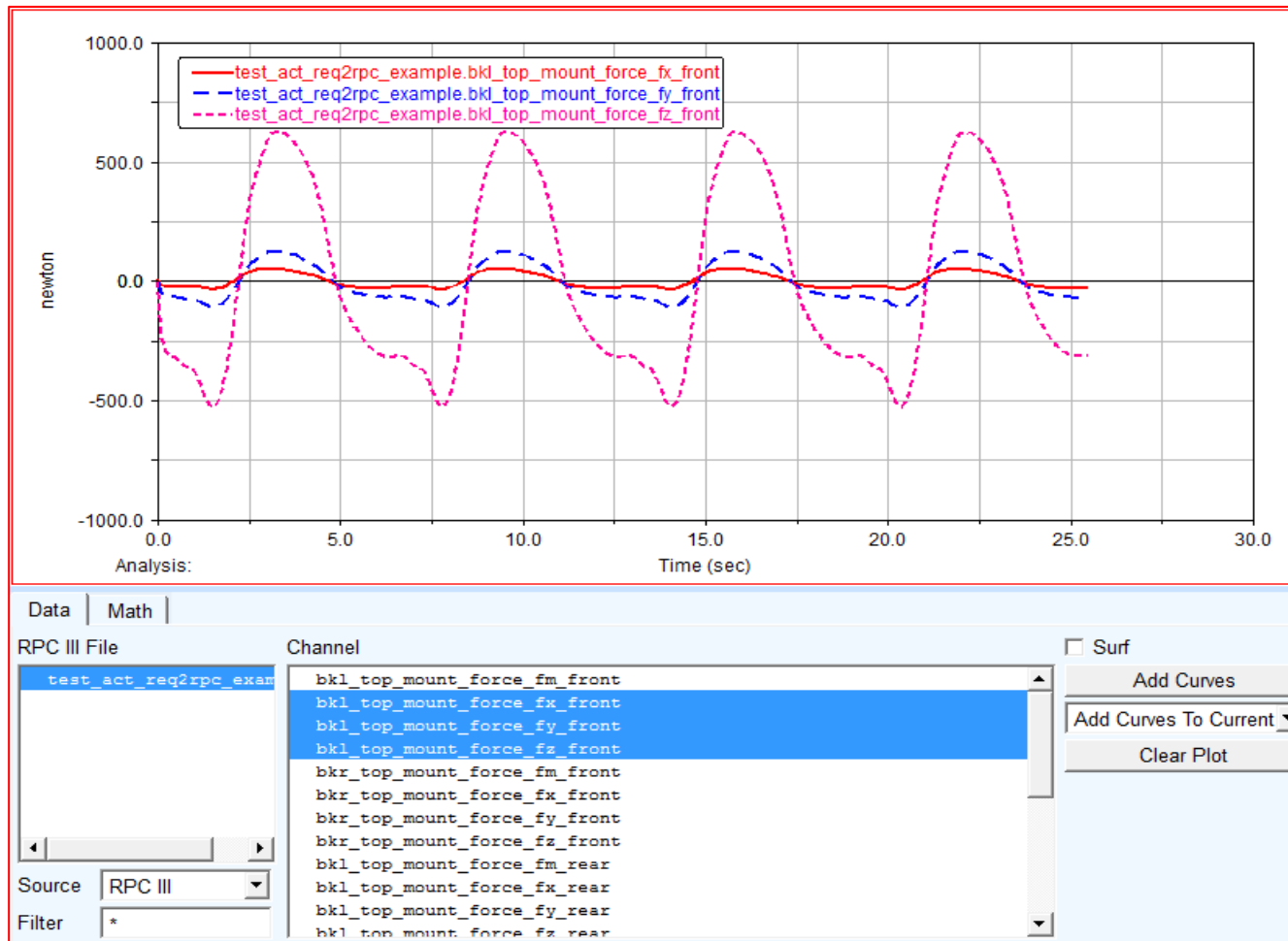
4. To import the RPC file, go to **File → Import → RPC File...**, right click in the **File to Read** text box and select **Browse**.
5. Select the **test_act_req2rsp_example.rsp** file generated as a part of simulation output in your working directory.



6. Select **OK**.

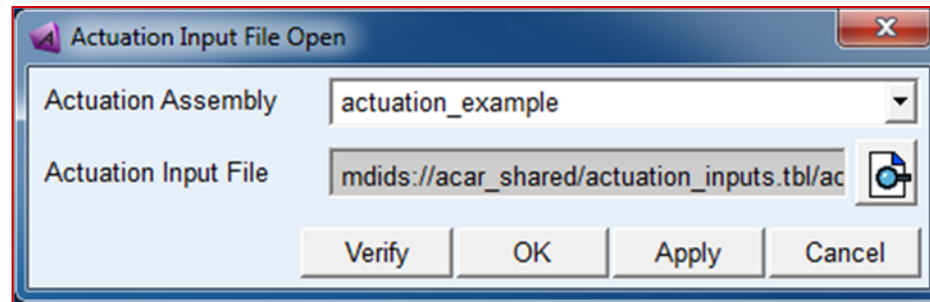
Plotting results in Adams Postprocessor

7. Select **bkl_top_mount_force_fx_front**, **bkl_top_mount_force_fy_front**, **bkl_top_mount_force_fz_front** under **channel** menu and select **Add Curves**.



Perform actuation analysis using RPC file

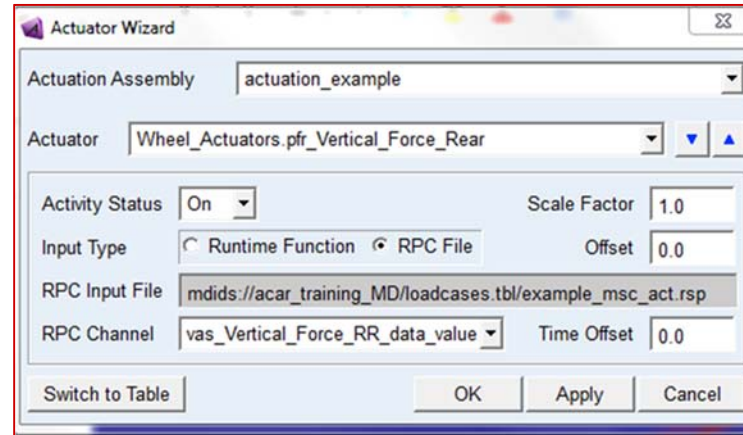
- To perform actuation analysis using RPC file:
 1. Go to **Simulate > GAA > Actuation Input File > Open**
 2. In the dialog box browse for the file **actuation_example.aci** from **acar_shared** database and
 3. Press **OK**.



4. Go to **Adjust > Actuators > Wizard**
5. Select **Wheel_Actuator.pfr_Vertical_Force_Rear** from the **Actuator** pull down menu.
6. Select **Input Type** as **RPC File**, and select **RPC Input File example_msc_act.rsp** from **acar_training_MD** database.

Perform actuation analysis using RPC file (Cont.)

7. Select **RPC Channel** as **vas_Vertical_Force_RR_data_value** and select **Apply**.



8. Similarly, repeat the steps through 5 to 7 for following actuators and select appropriate RPC Channel.

Actuator

Wheel_Actuator.pfl_Vertical_Force_Rear
Wheel_Actuator.pfr_Vertical_Force_Front
Wheel_Actuator.pfl_Vertical_Force_Front

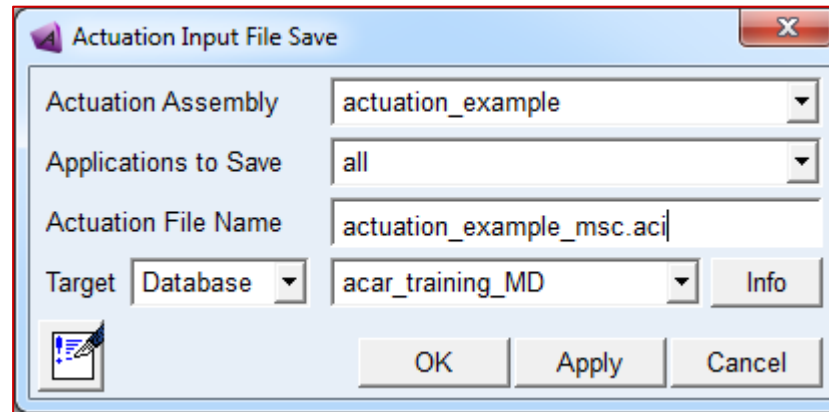
RPC Channel

vas_Vertical_Force_RL_data_value
vas_Vertical_Force_FR_data_value
vas_Vertical_Force_FL_data_value

Note: You can use up/down arrow to access different actuators.

Perform actuation analysis using RPC file (Cont.)

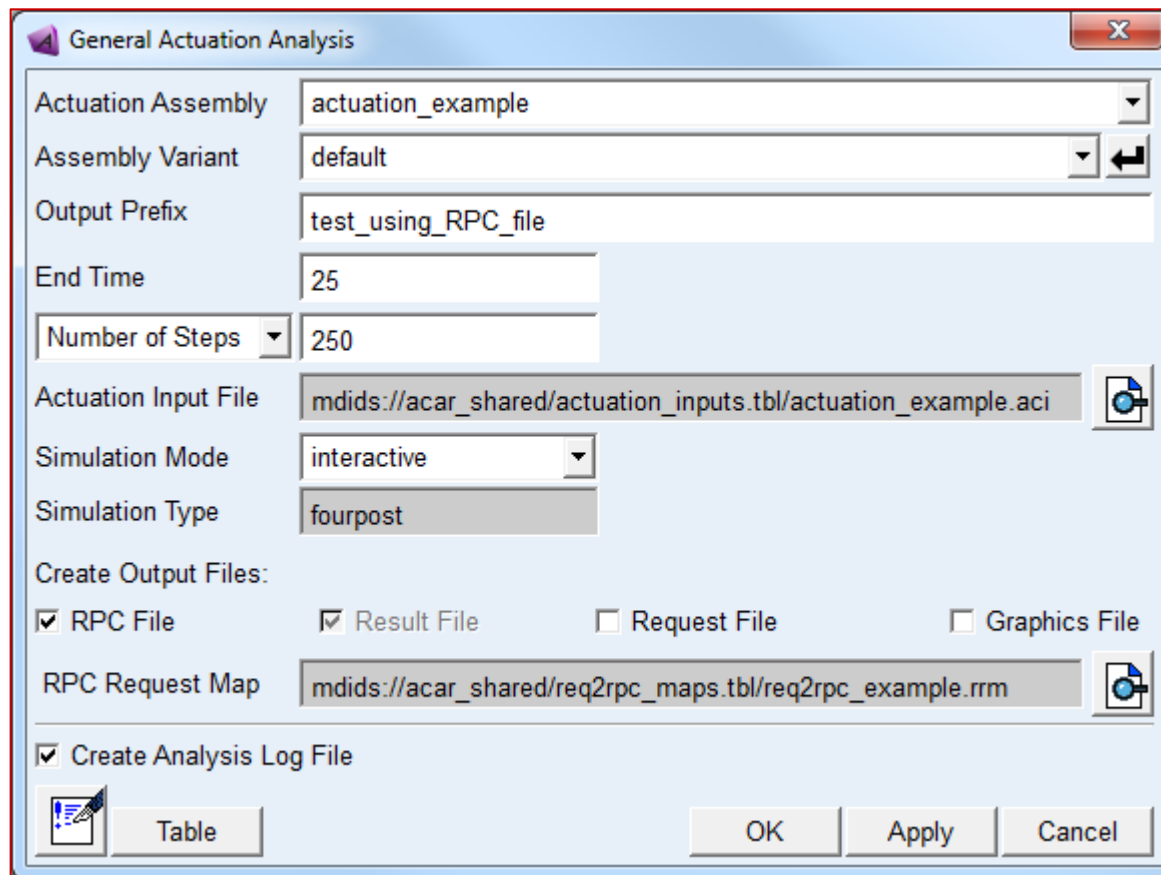
9. Go to **Simulate > GAA > Actuation Input File > Save**, Select **acar_training_MD** database as Target Database, Select **OK**.



10. Go to **Simulate > GAA > Submit Analysis**.
11. Fill the fields in the dialog box as shown on next slide
12. Select the Actuation Input File **actuation_example_msc.aci** from **acar_training_MD** database you just saved.
13. Select the RPC request map file **req2rpc_example.rrm** from **acar_shared** database.

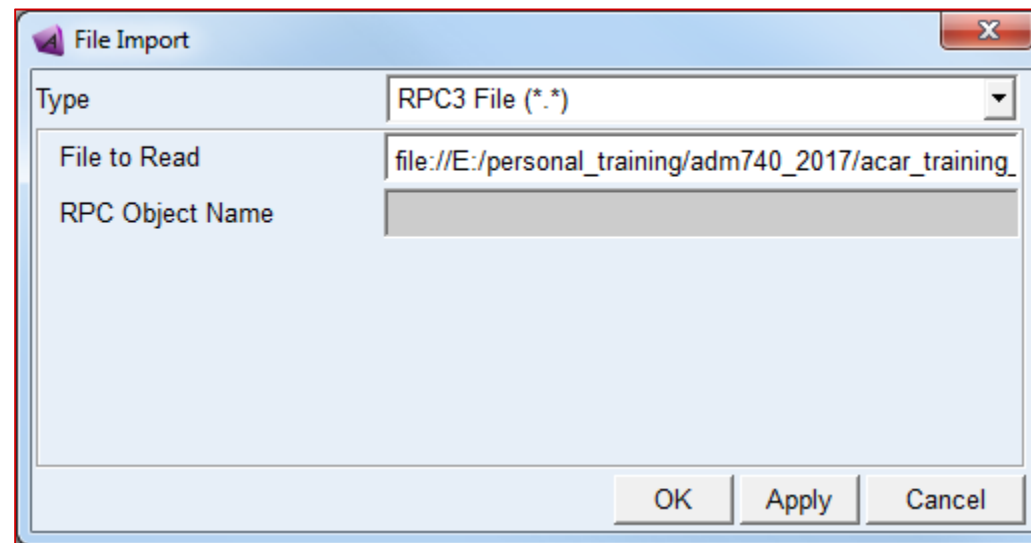
Perform actuation analysis using RPC file (Cont.)

14. Make sure the checkbox for **RPC File** and **Result File** are selected.
15. Select **OK**.



Perform actuation analysis using RPC file (Cont.)

- After completion of Simulation go to Adams/Postprocessor and verify that the RPC actuator inputs are indeed taken.
 1. To import the RPC file, go to **File → Import → RPC File...**
 2. Right click in the **File to Read** text box and Select the **example_msc_act.rsp** file from **acar_training_MD** database



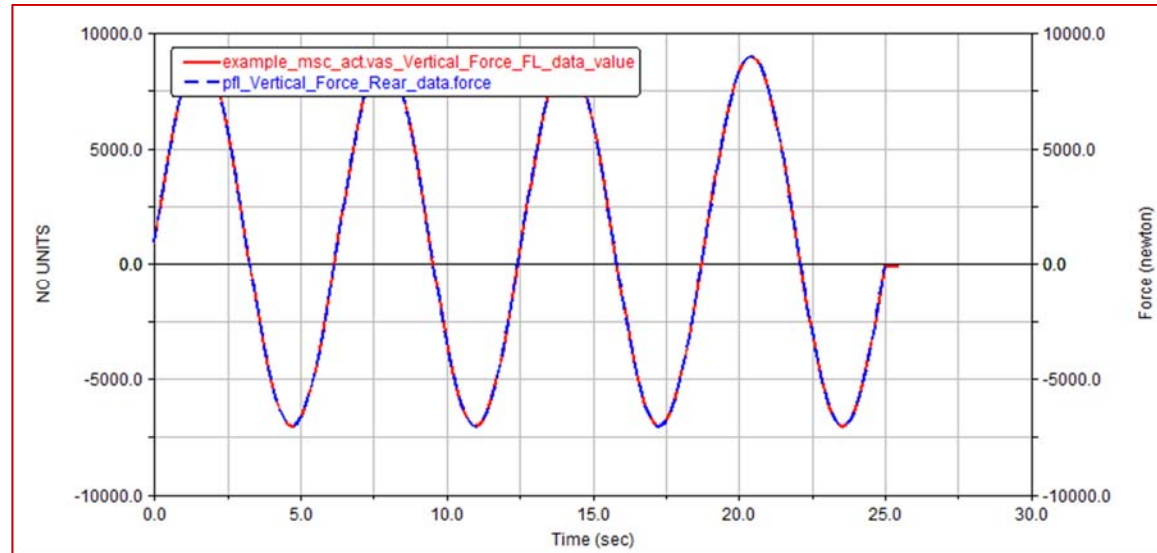
3. Select **OK**.

Perform actuation analysis using RPC file (Cont.)

4. Cross-plot the two curves:

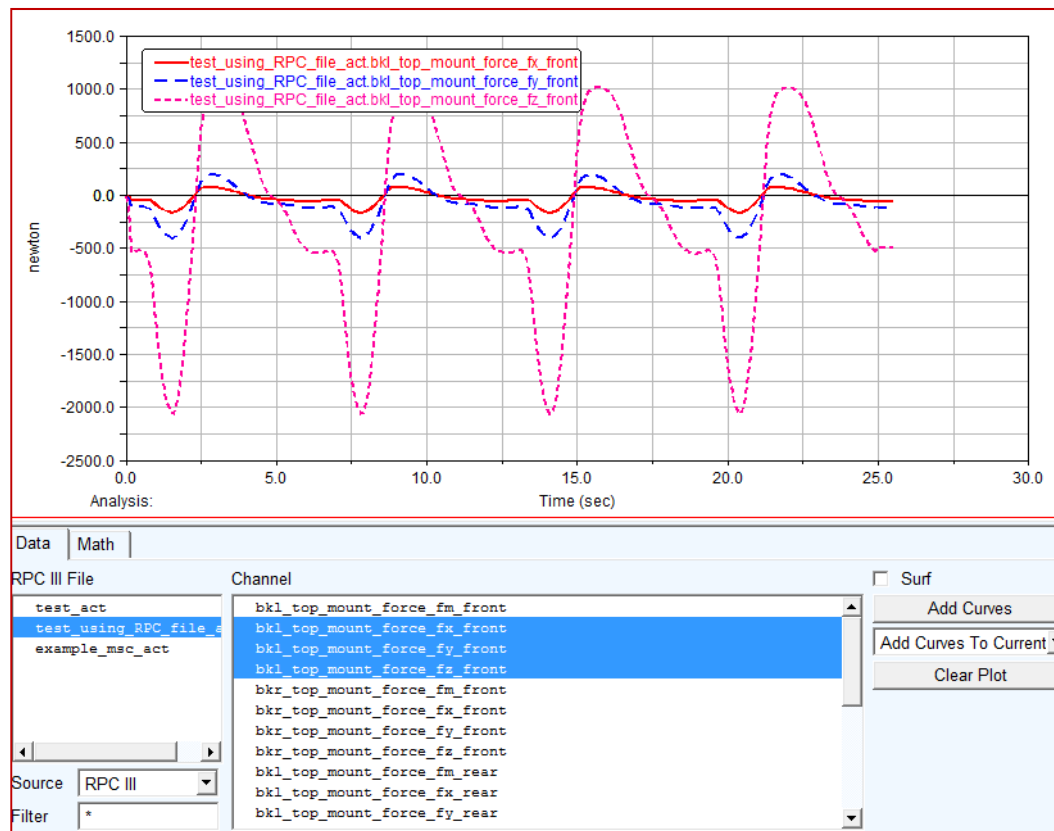
- Select **RPC III** from Source, **example_msc_act** under RPC III and **vas_Vertical_Force_RL_data_value** under channel, select **Add Curves**.
- Select **Requests** from Source, **test_using_RPC_file_act** under simulation, **pfl_Vertical_Force_Rear_data** under Request, select **Force**, select **Add Curves**.

The two curves should match with each other. This shows that the RPC actuator inputs are taken.



Perform actuation analysis using RPC file (Cont.)

- Select **RPC III** from Source menu, select **test_using_RPC_file_act** under RPC III menu, select **bkl_top_mount_force_fx_front**, **bkl_top_mount_force_fy_front**, **bkl_top_mount_force_fz_front** under channel menu and select **Add Curves**.





WORKSHOP 12

TEMPLATE-BUILDER TUTORIAL

Template-Builder Tutorial

- Go through Adams Online help → Getting Started → Getting Started Using Adams Car → Template Builder Tutorial.
- As you go through the tutorial, keep in mind that when building new templates, good practice is to write documentation about the templates. Otherwise, it will be difficult for someone else to use your templates.



This workshop takes about two hours to complete.

WORKSHOP 13

GETTING INFORMATION ABOUT COMMUNICATORS

Getting Information about Communicators

- **Problem statement:**

- In this workshop, you will learn how to perform tests and display information that will help you understand communicators.



This workshop takes one hour to complete.

Getting Information about Communicators

- **Opening a template**

To open a template:

1. From the **File** menu, select **Open**.
2. Right-click the **Template Name** text box, point to **Search**, and then select **<acar_shared>\templates.tbl**.
3. Double-click **rack_pinion_steering.tpl**.
4. Select **OK**.

The rack and pinion steering model appears in your workspace.

Getting Information about Communicators

- **Getting communicator information**

- To get information about a communicator in your template:

1. From the **Build** menu, point to **Communicator**, and then select **Info**.

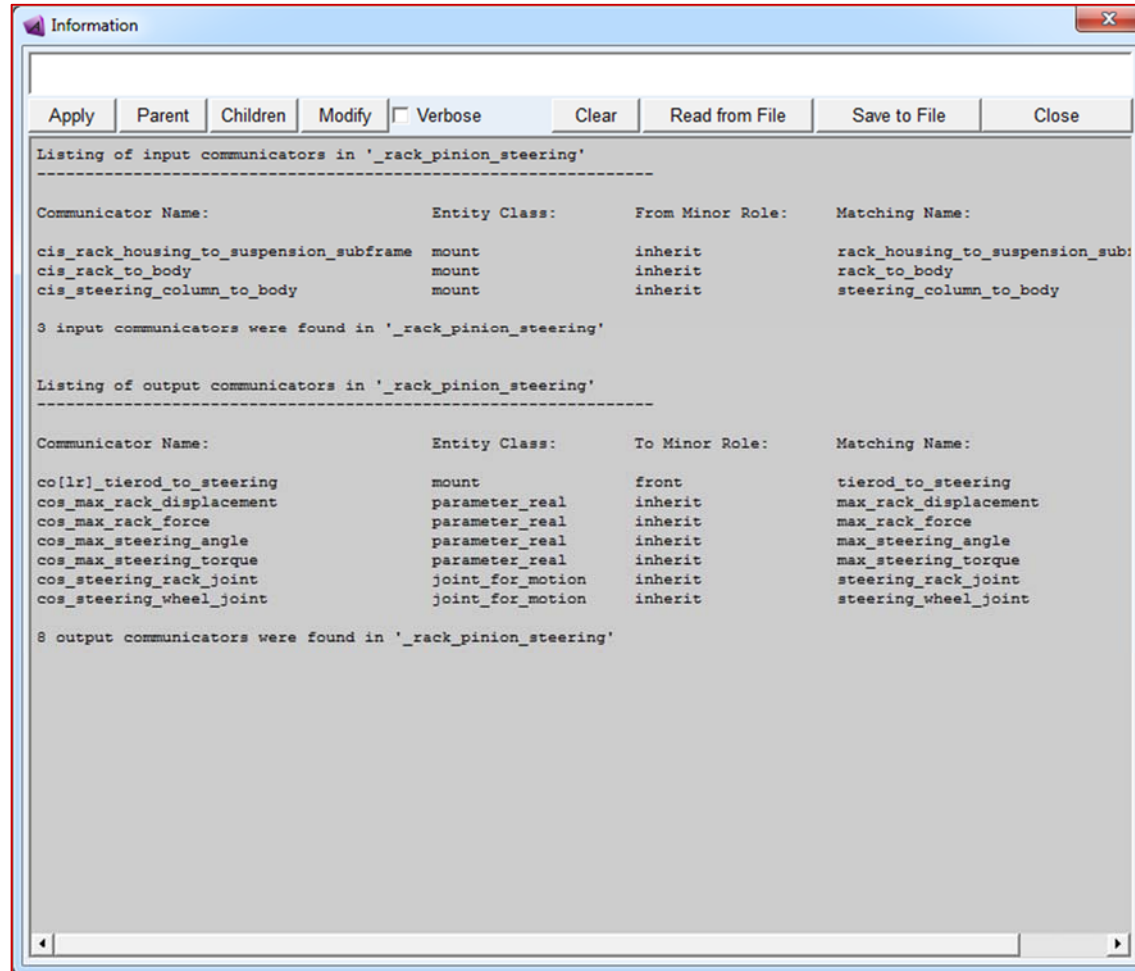
Note: The Model Names text box automatically displays the name of the template in the active view.

- **To list information about all the communicators in your template:**

2. In the **Communicators Info** dialog box, set **Entity** to **All**.
3. Select **OK**.

Getting Information about Communicators

The Information window lists information for all communicators in the rack and pinion template:



Getting Information about Communicators

- **Checking communicators**

- In this section, you will set up your workspace such that you can see how the communicators in your rack and pinion template match up with those in a suspension template. You will open a suspension template that your steering template will be connected to when you run analyses. By getting information about communicators, you will be able to determine what you must do (for example, what communicators you must create) for your templates to assemble correctly.

Note: When testing communicators, make sure that all templates you want to test are open in your Adams/Car session.

Getting Information about Communicators

- **To open a template:**

1. From the **File** menu, select **Open**.
2. Right-click the **Template Name** text box, point to **Search**, and then select **<acar_shared>\templates.tbl**.
3. Double-click **_double_wishbone_torsion.tpl**.
4. Select **OK**.

The template opens as a new model that is not associated with your rack and pinion template, which is still open.

The double-wishbone suspension appears in your workspace.

- **To perform a communicator test:**

1. From the **Build** menu, point to **Communicator**, and then select **Test**.
2. Clear the **Model Names** text box.
3. Right-click the **Model Names** text box, point to **Model**, point to **Guesses**, and then select **_double_wishbone_torsion**.
4. Repeat Step 3 to select **_rack_pinion_steering**.

Getting Information about Communicators

5. In the **Minor Roles** text box, enter the minor roles of the communicators (for example, front). You must enter one minor role for each model or test rig that you select.

Note: Each communicator has a minor role, which by default is one of the following: any, front, rear, trailer, or inherit. The inherit minor role specifies that when Adams/Car creates a subsystem from the template, the communicator should inherit the subsystem's minor role. Because when you test a template's communicators, the inherit minor role is still undefined, entering minor roles in the Minor Role text area provides the communicators with their minor role. For example, if you assign the template susp_02, a minor role of front in the Minor Roles text area, the communicator test also changes the minor role of any communicators in susp_02 whose minor role is inherit to the role of front.

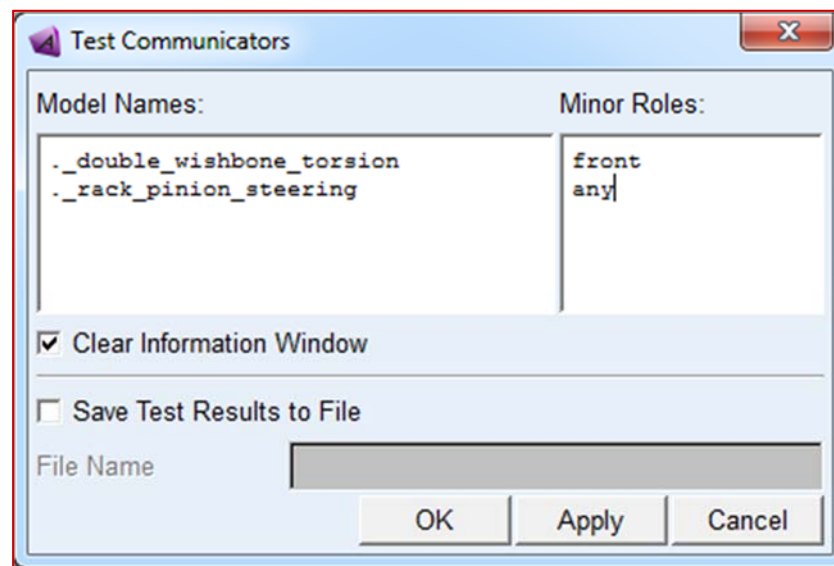
6. If you've used the Information window before, select **Clear Information Window**.

Getting Information about Communicators

7. If you want to save the results, select **Save Test Results to File**.
8. In the **File Name** text box, enter a file name.

For example, the filling out the box as shown below tests the communication between the suspension template with role set to **front** and the steering template with role set to **any**.

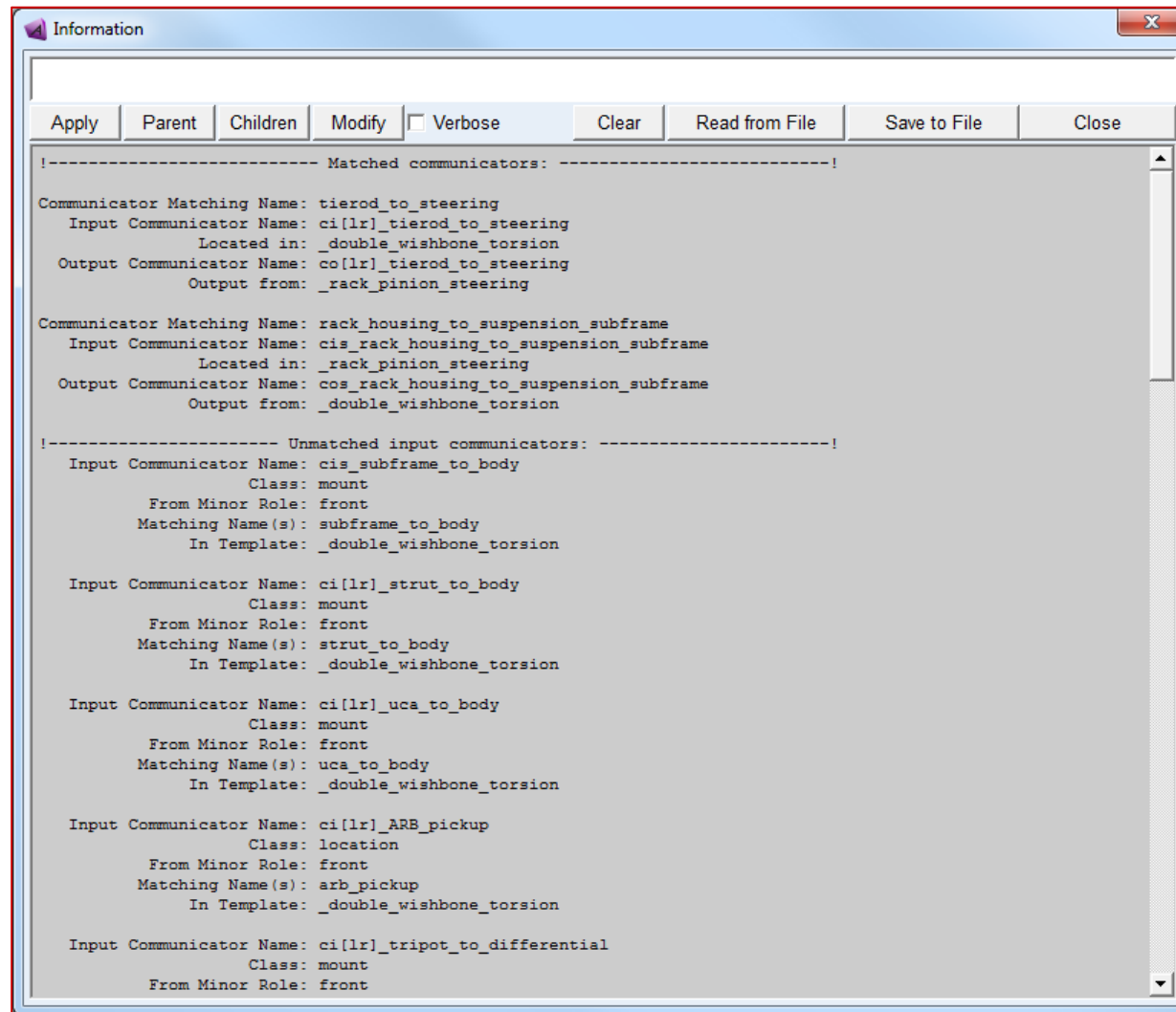
9. Select **OK**.



Getting Information about Communicators

- The Information window, as shown next, lists the communicators that match other communicators and those that do not. It shows the matched communicators followed by the unmatched communicators. The lists include the names of the input and output communicators and the names of the templates to which they belong. Often, you'll see many communicators that are unmatched. Many of these communicators are related to subsystems or test rigs that you do not currently have open.
- In this case, note that the communicators **rack_housing_to_suspension_subframe** and **tierod_to_steering** are matched. However, communicators such as **rack_to_body**, and **max_rack_force** are unmatched, because the particular templates which require this information were not selected (and unopened).

Getting Information about Communicators



Getting Information about Communicators

- To get to know the topology of the communicators within a template, study the inputs and outputs and look to see how they link with other templates. When you create communicators in a new template, a good way to create the communicators is to open an existing template and create the same communicators.
- A final tip is to remember that output communicators publish information. Therefore, they are passive and do not affect subsystem behavior. However, input communicators search for information. They affect your simulation, particularly if they do not match.

Location and Mount Communicators

- To assemble your templates, there are two communicator "entity" types that you primarily use such as
 - Mount
 - Location
- **Mount type communicator:**
 - A mount communicator associated with a mount part, which is a temporary part **use as a placeholder in an attachment** (joint or bushing) that will be used to connect one template to another.
 - The mount part and mount communicator allows you to **connect parts together (topologically) between subsystems with attachments** (e.g., joints or bushings).
 - Note that the parts don't have to be next to each other to connect them, nor does this communicator position the parts. The parts could be 100 yards away from each other and still be connected dynamically after the communicators connect them via the mount part.

Location and Mount Communicators

- **Location type communicator:**

- A location communicator contains a single location (x_coord, y_coord, z_coord), and this can be used to move parts around relative to another part. This allows you to position anything that depends on a location.
- For example, you can use a location communicator to specify the location of a construction frame, and in turn, this construction frame defines the location of a part. Thus, the part moves according to the construction frame which moves according to the communicator.

Note: Simply positioning parts next to each other doesn't mean that parts are connected by an attachment. This is the purpose of the mount part and mount communicators.

Location and Mount Communicators

- **In Summary:**

- Mount communicators connect templates via the parts. This must be done in order to assemble your subsystems properly.
- Location communicators simply locate other entities. These are not necessary, but could affect your results if one of your subsystems is not in an appropriate location.
- Note that you can always "shift" (Adjust --> Shift) subsystems or redefine the location of specific elements one by one to specify the correct location for entire subsystems or parts. The location communicator is simply a method to automatically locate elements.

- **For more information on mount and location communicators, see the following SimCompanion articles:**

- Article KB8014797 -- Example of use of location communicators at:
<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8014797>
- Article KB8014798 -- Example of use of mount communicators at:
<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8014798>

Optional Task

- **BUILDING A WHEEL TEMPLATE**

Building a Wheel Template (Optional Task)

- In this workshop, you will create a wheel template that sets up the interface to your tire model.



This workshop takes about one hour to complete.

Building a Wheel Template (Optional Task)

- **Creating a template**

To create the wheel template:

1. From the **File** menu, select **New**.
2. Name the template **wheel**.
3. Set **Major Role** to **wheel**.
4. Select **OK**.

Building a Wheel Template (Optional Task)

- **Creating communicators**

1. Create two left/right input communicators for toe and camber angles. They should be of type **parameter real**, have the role **inherit**, and be named **toe_angle** and **camber_angle**.

Note: Each communicator created for the wheel template should be set to the role **inherit**, so that when a wheel subsystem is created, each communicator will be present and have its inherited role. Is this always the case (for other types of templates)?

2. Create a left/right **input location communicator**, named **wheel_center**, that receives the wheel center location. This communicator positions the wheel at the location dictated by the suspension. The corresponding output communicator, **wheel_center**, resides in the MacPherson suspension you created in

Tip: Enter an initial value for the wheel center, or else the default value will be 0,0,0, and the wheel centers will be located on top of each other. You will see the result of this in the next steps. Remember that the negative y-direction is left.

Building a Wheel Template (Optional Task)

- **Creating construction frames and mount parts**

To create the construction frame and mount part:

1. Create a left/right **construction frame**, named **wheel_center**, to be used as the spin axis. Locate this construction frame on the **wheel_center input communicator**, which should match the output communicator from the suspension. This construction frame should also be dependent on two communicators for toe and camber angles. (See the different orientation dependency options.) Verify that the spin axis is defined correctly, such that the z-axis points out from the vehicle (this should be done automatically for you).
2. For the wheel, create the **mount part** that will attach to the suspension, and name the mount part, **wheel_to_susp**. The coordinate reference should be the **wheel center** construction frame.

Building a Wheel Template (Optional Task)

- **Creating the wheel part**

To create the wheel part:

1. From the **Build** menu, point to **Wheel**, and then select **New**.
2. Create the wheel part with the following data:
 - **Name:** wheel
 - **Mass:** 20.0 (kg)
 - **Ixx Iyy:** 5E4 (kg-mm**2)
 - **Izz:** 1E4 (kg-mm**2)
 - **Property File:** mdi_tire01.tir
 - **Coordinate Reference:** cfl_wheel_center
 - **Location:** 0, 0, 0 mm
 - **Construction Frame:** cfl_wheel_center
 - **Orientation:** 0, 0, 0 (deg)
3. Select **OK** to create the wheel part.

Note: Adams/Car automatically creates a pair and sets the tire geometry in the property file.

Building a Wheel Template (Optional Task)

- **Viewing tire geometry**

To view the tire geometry values:

1. To view the tire property file, open the Modify Wheel dialog box and select the **View File** tool .
2. Search for the data block named **DIMENSION**.

The parameters in this data block specify the size of the tire geometry.

- **Connecting mount and wheel parts**

To connect the mount part to the wheel part:

1. Create a fixed joint between the wheel and the mount part located at the **cfl_wheel_center**.

Building a Wheel Template (Optional Task)

- **Modifying communicators**

- You'll now modify a mount output communicator in the MacPherson template to be able to assemble these two templates as subsystems.

- **To modify a communicator:**

1. Open the MacPherson template you created in Workshop 11—Template-Builder Tutorial in Adams Online help (be sure to use the one you created rather than the one in the shared database, because these have topological differences).
2. From the **Build** menu, point to **Communicator**, point to **Output**, and then select **Modify**.
3. Modify **co[l,r]_suspension_mount** to include the matching name **wheel_to_susp**.

This allows the MacPherson template to pass on information to the wheel template about the part to which the wheels should attach.

Building a Wheel Template (Optional Task)

- **Testing communicators**

To test communicators:

- Test the communicators and select this wheel template and the suspension template you intend to use (make sure that you have these templates open in your session).
- Note that this template has been developed to be used with the MacPherson suspension template you created in (If you use this template with another suspension, there might be other communicators you must match).

Building a Wheel Template (Optional Task)

- **Testing assemblies**

To check communicators:

1. Save the templates for the generic assembly.
2. Switch to Standard Interface, and create subsystems for your templates.
3. Save the subsystems you created in the previous step.
4. Create the generic assembly (for testing only, since you're not including a test rig):
 - **File → new → Generic Assembly...**
 - **Test Rig : Un-check**
 - **assembly class: generic**
 - **subsystems: wheel and macpherson**
5. Investigate the assembly to confirm that the subsystems assembled correctly.

Building a Wheel Template (Optional Task)

Note: The tire template you just created contains the minimal entities to run with Adams/Car and Adams/Tire. Use the tire template (`_handling_tire.tpl`) in the shared Adams/Car database if you want to use all the features of Adams/Car (for example, quasi-static setup), unless you require a special template. Some of the other entities set up in the `_handling_tire.tpl` model are:

- A pair of DIFFs, one for each tire. Static-setup analyses use the DIFFs to determine the initial longitudinal slip of the tires. The tire UDE parameters reference the DIFF.
- A pair of JPRIMs (`jo[lr]per_sse_jprim`) to lock the wheel during the static-setup analysis. These are perpendicular JPRIMs between the wheel part and ground (road). Tire UDEs reference these JPRIMs.
- Output communicators `co[lr]_tire_force` to output the left and right tire force Adams_ids to powertrain templates for wheel omega calculations.

WORKSHOP 14

REQUESTS

Requests

- **Problem Statement:**

- In this workshop you will create three types of requests:

- Define Using Type And Markers
 - Define Using Function Expression
 - Define Using Subroutine

- **In this workshop, you will:**

- Build a request in a template
 - Build a subsystem and assembly using the template
 - Plot the request after running a suspension analysis on a suspension assembly

- **If time permits optional task of using communicators for creating requests between templates.**



This workshop takes about one and a half hours to complete.

Requests

- **To Open An Existing Template:**
 1. Launch Adams/Car in Template Builder Mode.
 2. From the **File** menu, point to **Open**.
 3. In the **Template Name** text box, point to **Search**, and then select **<acar_shared>\templates.tbl**.
 4. Double-click **_double_wishbone.tpl**.
 5. Select **OK**.

Adams/Car displays the template

Requests

- **Define Using Type and Markers:**

- You could request four types of output such as displacement, velocity, acceleration and force. In this workshop, you will use displacement type of output to create a new request to measure wheel travel.

To create a new request:

1. From the **Build** menu, point to **Request**, and then select **New**.
2. In the **Request Name** text box, enter **my_wheel_travel_using_markers**.
3. In the **Comment** text box, enter **my_wheel_travel_using_markers**.
4. Select **Define Using Type And Markers**.
5. Select **displacement as Output Type**
6. In the **I Marker Name** text box, enter.
_double_wishbone.gel_spindle.cm

Requests

- **Define Using Type and Markers (Cont.):**
 7. In the J Marker Name text box, enter **._double_wishbone.ground.cfl_wheel_center**
 8. In the R Marker Name text box, enter **._double_wishbone.ground.origo**
 9. In the Result Set Name text box, type name for result, **my_wheel_travel_using_markers**.
 10. Enter the strings that correspond to the output requests and identify the components as shown in the dialog box.
 11. Select **OK**.

Note: This request is for left wheel travel; you could create a similar request for right wheel travel.

Component Attributes

Component	Attribute	Value
MAG	dm	length
X	dx	length
Y	dy	length
Z	dz	length
AMAG	amag	angle
R1	ax	angle
R2	ay	angle
R3	az	angle

Requests

- **Define Using Function Expression:**

- You could request any type of outputs by using runtime functions like SFORCE, GFORCE, VZ, VM, DZ, DM, etc. In this workshop, you will use 'DZ' displacement type of output to create new request to measure wheel travel.

To create a new request:

1. From the **Build** menu, point to **Request**, and then select **New**.
2. In the **Request Name** text box, enter **my_wheel_travel_using_fun_exp**.
3. In the **Comment** text box, enter **my_wheel_travel_using_fun_exp**.
4. Select **Define Using Function Expression**.
5. In the **F2** text box, enter function expression to measure left wheel travel
**DZ(._double_wishbone.gel_spindle.cm,
._double_wishbone.ground.cfl_wheel_center,
._double_wishbone.ground.origo)**

Requests

- **Define Using Function Expression (Cont.):**

6. In the **F3** text box, enter function expression to measure right wheel travel
DZ(._double_wishbone.ger_spindle.cm, . _double_wishbone.ground.cfr_wheel_center, . _double_wishbone.ground.origo)
7. In the **Result Set Name** text box, type name for result,
my_wheel_travel_using_fun_exp.
8. Enter the strings that correspond to the output requests and identify the components as shown in the dialog box.
9. Select **OK**.

Component	Value	Units
MAG		no_units
X	wheel_travel_left	length
Y	wheel_travel_right	length
Z		no_units
AMAG		no_units
R1		no_units
R2		no_units
R3		no_units

Requests

- **Define Using Subroutine:**

- You could use user defined subroutine to request any type of outputs. In general you could create all types of requests using function expression instead of user subroutine. In this workshop, you will use Adams/Car standard subroutine 'req905' to request force acting on lower_control_arm front bushing.

To create a new request:

1. From the **Build** menu, point to **Request**, and then select **New**.
2. In the **Request Name** text box, enter **my_force_request_using_subroutine**.
3. In the **Comment** text box, enter **my_force_request_using_subroutine**.
4. Select **Define Using Subroutine**.
5. In the **User Function** text box, enter parameter required for user subroutine: **905.0, 3.0,**
(.double_wishbone.bkl_lca_front.i_marker[1].adams_id),
(.double_wishbone.bkl_lca_front.j_marker[1].adams_id),
(.double_wishbone.bkl_lca_front.field.adams_id)

Requests

- **Define Using Function Expression (Cont.):**

6. In the **Routine** text box, enter library name and subroutine name **abgFDM::req905**
7. In the **Result Set Name** text box, type name for result, **my_force_request_using_subroutine**.
8. Enter the strings that correspond to the output requests and identify the components as shown in the dialog box.
9. Select **OK**.

Create Request

Request Name: my_force_request_using_subroutine

Comment: my_force_request_using_subroutine

Define Using Subroutine: [dropdown]

User Function: 905.0, 3.0, (, _double_wishbone.bkl_ica_front_i_r

Title: [empty]

Routine: abgFDM::req905

Component Attributes

	Result Set Name	
MAG	fm	force
X	fx	force
Y	fy	force
Z	fz	force
AMAG	tm	torque
R1	tx	torque
R2	ty	torque
R3	tz	torque

OK Apply Cancel

Requests

- **To Create Group For Request:**

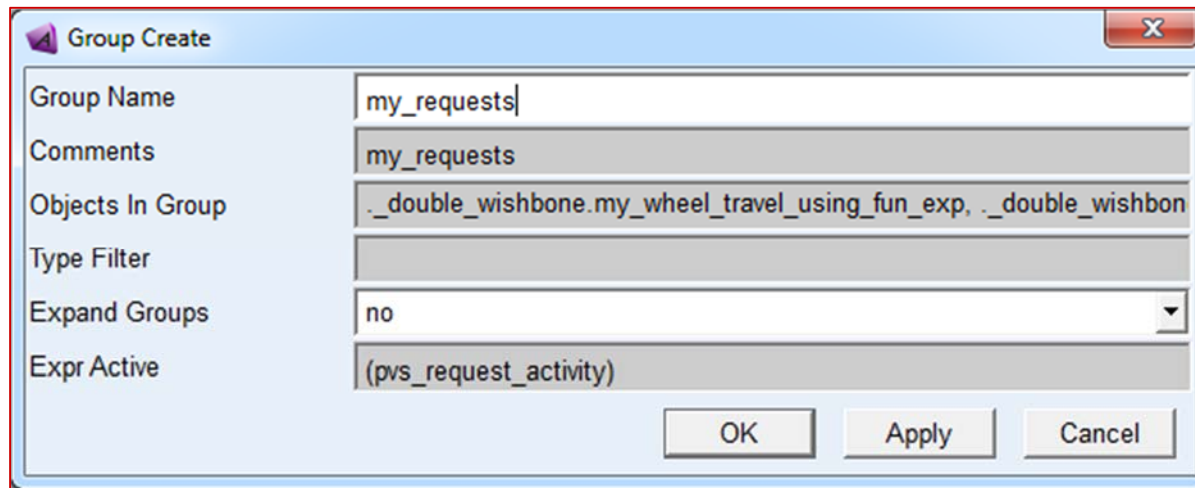
- You will create a group to group newly created request. You will create parameter that could be used to toggle those request.
 1. From the **Build** menu, point to **Parameter Variable** then **New**
 2. In the **Parameter Variable Name** test box, enter **request_activity**.
 3. Set **type** to **single**.
 4. Enter **1.0** as the **Real Value**.
 5. Select **OK**.
 6. From the **Tool** menu, point to **Command Navigator**, and then expand **group** to select **create**.
 7. In the **Group Name** text box, enter **my_requests**.
 8. In the **Comments** text box, enter **my_requests**.
 9. In the **Objects In Group** text box, point to **Browse**, and then select request like
._double_wishbone.my_wheel_travel_using_fun_exp,
._double_wishbone.my_wheel_travel_using_markers,
._double_wishbone.my_force_request_using_subroutine

Requests

- **To Create Group For Request (Cont.):**

10. In the **Expr Active** text box, enter **pvs_request_activity**.

11. Select **OK**.



The screenshot shows a 'Group Create' dialog box. The 'Group Name' field contains 'my_requests'. The 'Comments' field contains 'my_requests'. The 'Objects In Group' field contains a list of objects: '_double_wishbone.my_wheel_travel_using_fun_exp, _double_wishbon'. The 'Type Filter' field is empty. The 'Expand Groups' dropdown menu is set to 'no'. The 'Expr Active' field contains '(pvs_request_activity)'. The 'OK', 'Apply', and 'Cancel' buttons are at the bottom right.

12. From **File** menu, select **Save As** and save this template in the **acar_training** database as **_double_wishbone.tpl**.

Requests

- **To Carry Out Suspension Analysis:**

- You create new suspension subsystem using saved template _double_wishbone.tpl file and carry out analysis for parallel wheel travel.
- To create Suspension Assembly:
 1. Change to Adams/Car Standard Interface.
 2. From the **File** menu, point to **New**, and then select **Subsystem**.
 3. In the **Subsystem Name** text box, enter **my_double_wishbone**.
 4. Set **Minor Role** to **front**.
 5. Right-click the **Template Name** text box, point to **Search**, and then select **<acar_training>\templates.tbl**.
 6. Double-click **_double_wishbone.tpl**.
 7. Select **OK**.

Adams/Car displays the subsystem.

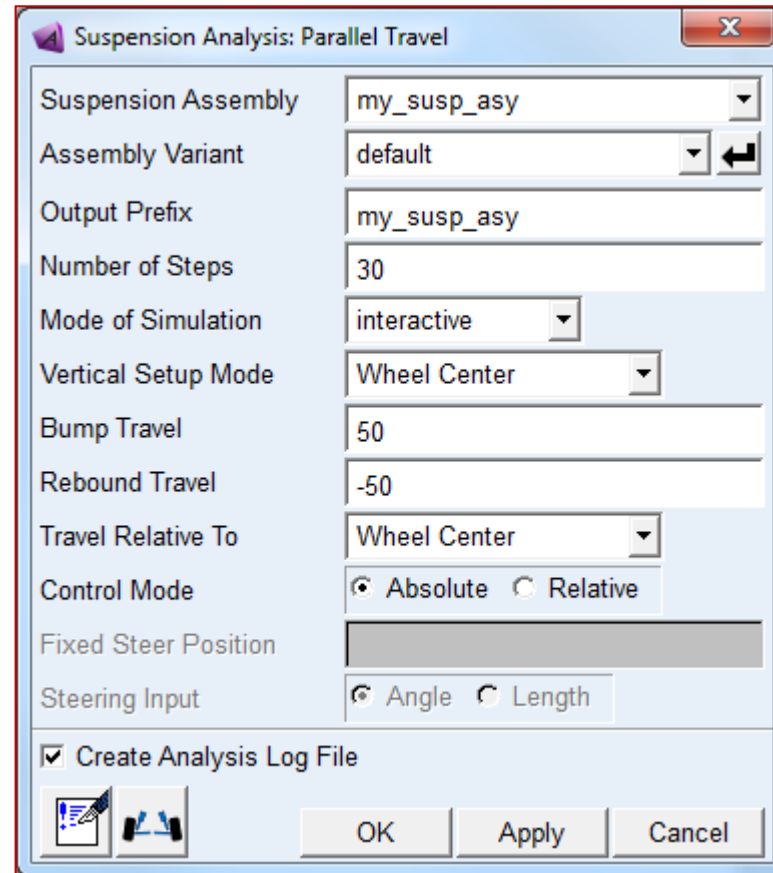
Requests

- **To Carry Out Suspension Analysis (Cont.):**

8. From the **File** menu, point to **Save As**, and then select **Subsystem**. Save subsystem file in the **acar_training** database.
9. From the **File** menu, point to **New**, and then select **Suspension Assembly**.
10. In the **Assembly Name** text box, enter **my_susp_asy**.
11. Right-click the **Suspension Subsystem** text box, point to **Search**, and then select **<acar_training>\subsystems.tbl**.
12. Double-click **my_double_wishbone.sub**
13. Select **OK**.
14. From the **Simulate** menu, point to **Suspension Analysis**, and then select **Parallel Wheel Travel**.

Requests

- **To Carry Out Suspension Analysis (Cont.):**
15. Enter values as shown in the dialog box.



16. Select **OK**.

Requests

- **To Carry Out Suspension Analysis (Cont.):**
 17. After completion of the simulation, press **F8** to go to Postprocessor.
 18. Plot the request that you have created.

Requests

- **Optional - Creating Request Between Templates:**
- **Problem Statement:**
 - In this workshop, you will create a request between templates using communicator and mount part. You will create a request to measure a spring deformation in which one part is in the other template.

Requests

- **Steps To Create Request Between Templates:**

- You will open suspension template (`_double_wishbone.tpl`) and identify two parts like `gel_lower_strut` and `mtl_strut_to_body`.
- Out of two parts, one should be mount part that would get replaced by the part in the other template through mount communicator.
- Create input communicator of type location.
- Create marker on the mount part using input communicator to parameterize location.
- Create a request using marker on mount part and other part.
- Open second template (`._rigid_chassis`)
- Create construction frame at the desired location.
- Create output communicator of type location. This communicator should match with the input communicator that you have created earlier.
- Do communicator test.

Requests

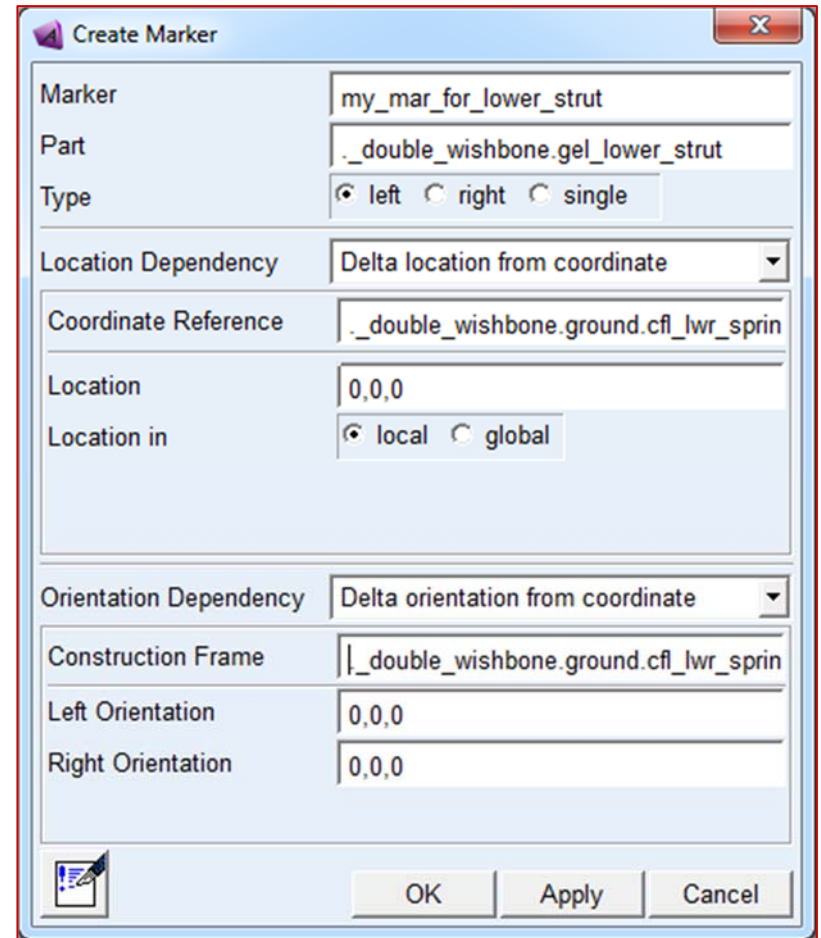
- **To Open First Template:**

1. Change to Adams/Car Template Builder Interface.
2. From the **File** menu, point to **Open**.
3. In the **Template Name** text box, point to **Search**, and then select **<acar_shared>\templates.tbl**.
4. Double-click **_double_wishbone.tpl**.
5. Select **OK**.

Adams/Car displays the template

Requests

- **To Create Markers and Input Communicator:**
 1. From the **Build** menu, point to **Marker** and then select **New**.
 2. In the **Marker** text box, enter **my_mar_for_lower_strut**
 3. In the **Part** text box, select left lower strut **gel_lower_strut**
 4. Enter rest of the values as per the dialog box
 5. Select **OK**.



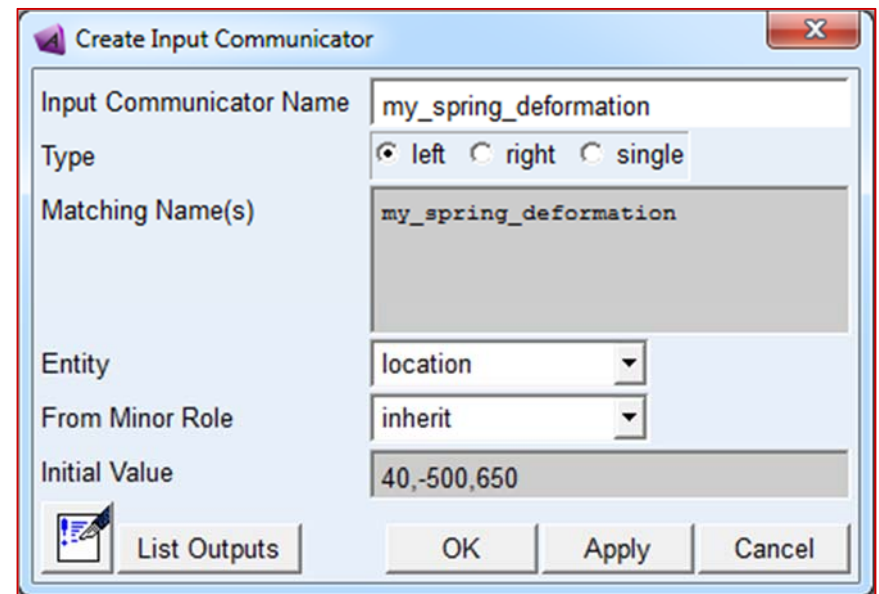
The screenshot shows the 'Create Marker' dialog box with the following fields and values:

Field	Value
Marker	my_mar_for_lower_strut
Part	._double_wishbone.gel_lower_strut
Type	<input checked="" type="radio"/> left <input type="radio"/> right <input type="radio"/> single
Location Dependency	Delta location from coordinate
Coordinate Reference	._double_wishbone.ground.cfl_lwr_sprin
Location	0,0,0
Location in	<input checked="" type="radio"/> local <input type="radio"/> global
Orientation Dependency	Delta orientation from coordinate
Construction Frame	._double_wishbone.ground.cfl_lwr_sprin
Left Orientation	0,0,0
Right Orientation	0,0,0

Buttons: OK, Apply, Cancel

Requests

- **To Create Markers and Input Communicator (Cont.):**
 6. From the **Build** menu, select **Communicator**, point to **Input** and then select **New**.
 7. In the **Input Communicator Name** text box, enter **my_spring_deformation**.
 8. In the **Matching Name** text box, enter **my_spring_deformation**.
 9. In the **Entity** pull down menu, select **location**.
 10. In the **Initial Value** text box, enter **any value**.
 11. Select **OK**.



Requests

- **To Create Markers and Input Communicator (Cont.):**

12. From the **Build** menu, point to **Marker** and then select **New**.

13. In the **Marker** text box, enter **my_marker_for_mount_part_strut_to_body**

14. In the **Part** text box, select left mount part **mtl_strut_to_body**

15. In the **Location Dependency**, select **Location Input Communicator**.

16. In the **Input Communicator**, enter **cil_my_spring_deformation**.

17. Select **OK**.

The screenshot shows the 'Create Marker' dialog box with the following fields and values:

- Marker:** my_marker_for_mount_part_strut_to_body
- Part:** _double_wishbone.mtl_strut_to_body
- Type:** ☒ left ☐ right ☐ single
- Location Dependency:** Location input communicator
- Input Communicator:** _double_wishbone.cil_my_spring_deformation
- Orientation Dependency:** User entered values
- Orient using:** ☒ Euler Angles ☐ Direction Vectors
- Euler Angles:** 0,0,0
- X Vector:** 1,0,0,0,0,0
- Z Vector:** 0,0,0,0,1,0

Buttons at the bottom: List Inputs, OK, Apply, Cancel.

Requests

- **To Create Request:**

1. From the **Build** menu, point to **Request**, and then select **New**.
2. In the **Request Name** text box, enter **my_spring_deformation**.
3. In the **Comment** text box, enter **spring_deformation**.
4. Select **Define Using Function Expression**.
5. In the **F2** text box, enter
DM(._double_wishbone.mtl_strut_to_body.mal_my_marker_for_mount_part_strut_to_body,._double_wishbone.gel_lower_strut.mal_my_mar_for_lower_strut)
6. In the **F3** text box, enter
DM(._double_wishbone.mtr_strut_to_body.mar_my_marker_for_mount_part_strut_to_body,._double_wishbone.ger_lower_strut.mar_my_mar_for_lower_strut).

Requests

- **To Create Request (Cont.):**
 7. In the **Result Set Name** text box, type name for result, **spring_deformation**.
 8. Enter the strings that correspond to the output requests and identify the components as shown in the dialog box.
 9. Select **OK**.

Create Request

Request Name: my_spring_deformation
Comment: spring_deformation

Define Using Function Expression

F1:
F2: DM(.double_wishbone.mtl_strut_to_body.mal_my_marker_for_
F3: DM(.double_wishbone.mtr_strut_to_body.mar_my_marker_for_
F4:
F5:
F6:
F7:
F8:
Title:

Component Attributes

Result Set Name: spring_deformation

MAG		no_units
X	vertical_spring_deformation_left	no_units
Y	vertical_spring_deformation_right	no_units
Z		no_units
AMAG		no_units
R1		no_units
R2		no_units
R3		no_units

OK Apply Cancel

Requests

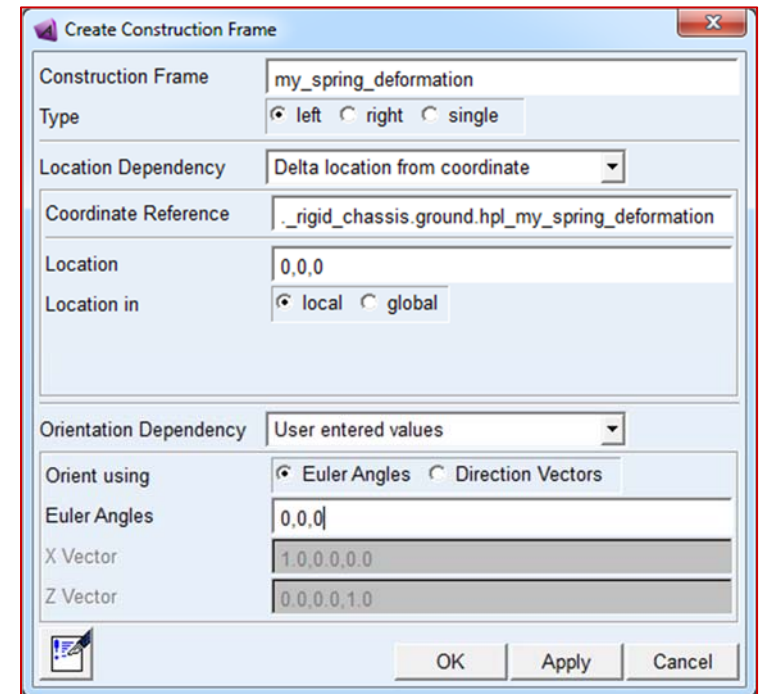
- **To Open Second Template:**

1. Change to Adams/Car Template Builder Interface.
2. From the **File** menu, point to **Open**.
3. In the **Template Name** text box, point to **Search**, and then select **<acar_shared>\templates.tbl**.
4. Double-click **_rigid_chassis.tpl**.
5. Select **OK**.

Adams/Car displays the template

Requests

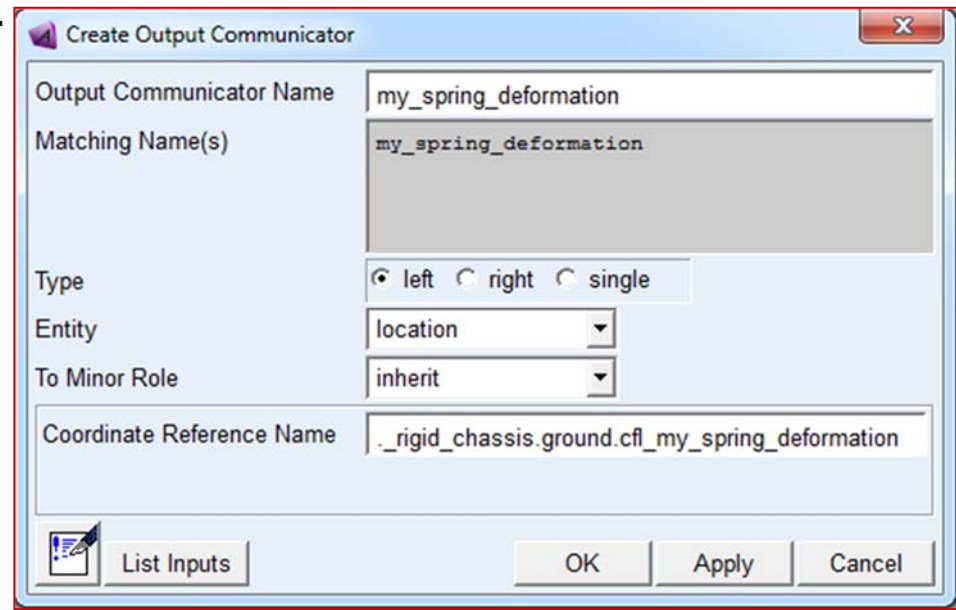
- **To Create Construction Frame and Output Communicator:**
 1. From the **Build** menu, point to **Hardpoint** and select **New**.
 2. In the **Name** text box, enter **my_spring_deformation**
 3. In the **location** text box, enter **307.0, -500.0, 680.0**
 4. Select **OK**.
 5. From the **Build** menu, point to **Construction Frame** and select **New**.
 6. Enter value as shown in the dialog box.
 7. Select **OK**.



Requests

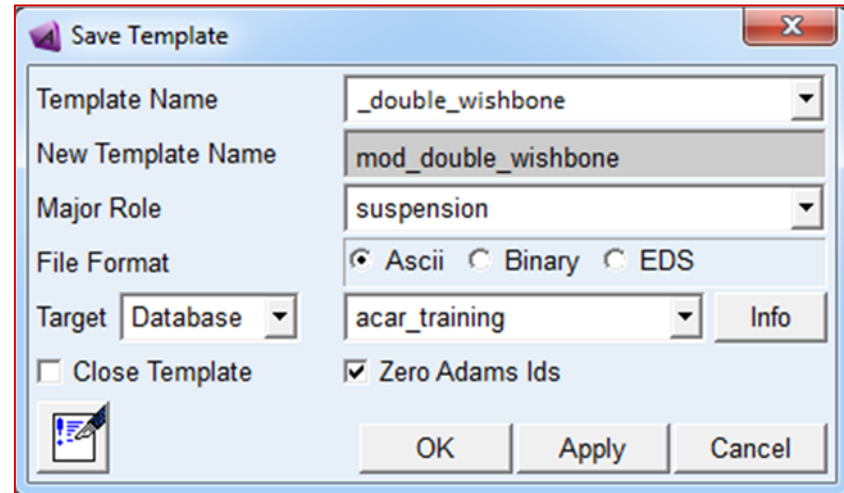
- **To Create Construction Frame and Output Communicator (Cont.):**

8. From the **Build** menu, select **Communicator**, point to **Ouput** and then select **New**.
9. In the **Ouput Communicator Name** text box, enter **my_spring_deformation**.
10. In the **Matching Name** text box, enter **my_spring_deformation**
11. In the **Entity** pull down menu, select **location**.
12. In the **Coordinate reference Name** text box, enter construction frame **cfl_my_spring_deformation**.
13. Select **OK**.



Requests

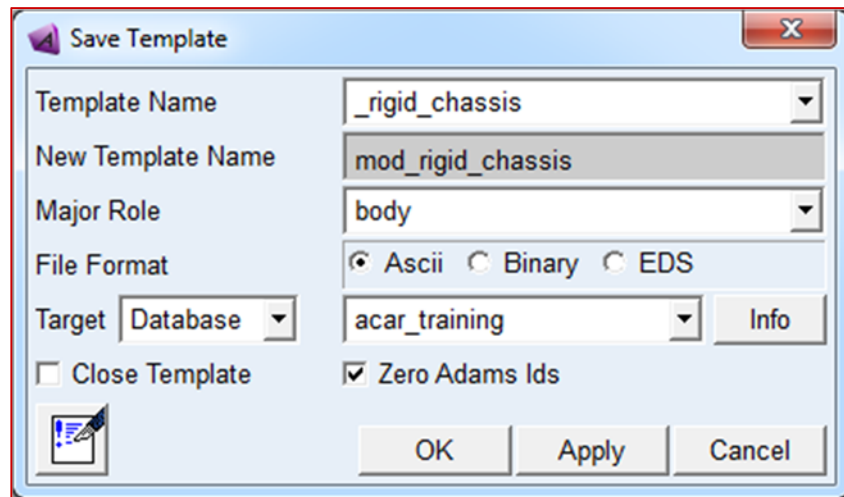
- **Saving the templates:**
 - Save the **double_wishbone** and **rigid_chassis** templates as (in **acar_training** database):
mod_double_wishbone.tpl
mod_rigid_chassis.tpl



The 'Save Template' dialog box for the double_wishbone template. It features a light blue background and a standard Windows-style title bar with a close button. The fields are as follows:

Field	Value
Template Name	_double_wishbone
New Template Name	mod_double_wishbone
Major Role	suspension
File Format	Ascii (selected), Binary, EDS
Target	Database (selected), acar_training
Close Template	<input type="checkbox"/>
Zero Adams Ids	<input checked="" type="checkbox"/>

Buttons at the bottom: OK, Apply, Cancel. An 'Info' button is next to the Target field.



The 'Save Template' dialog box for the rigid_chassis template. It features a light blue background and a standard Windows-style title bar with a close button. The fields are as follows:

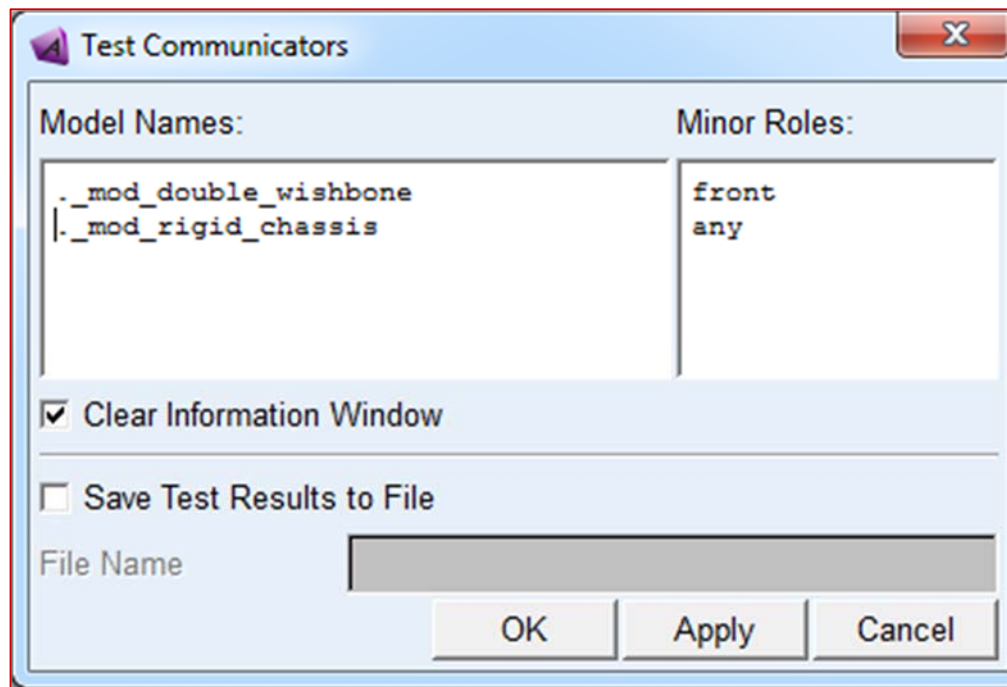
Field	Value
Template Name	_rigid_chassis
New Template Name	mod_rigid_chassis
Major Role	body
File Format	Ascii (selected), Binary, EDS
Target	Database (selected), acar_training
Close Template	<input type="checkbox"/>
Zero Adams Ids	<input checked="" type="checkbox"/>

Buttons at the bottom: OK, Apply, Cancel. An 'Info' button is next to the Target field.

Requests

- **Communicator test:**

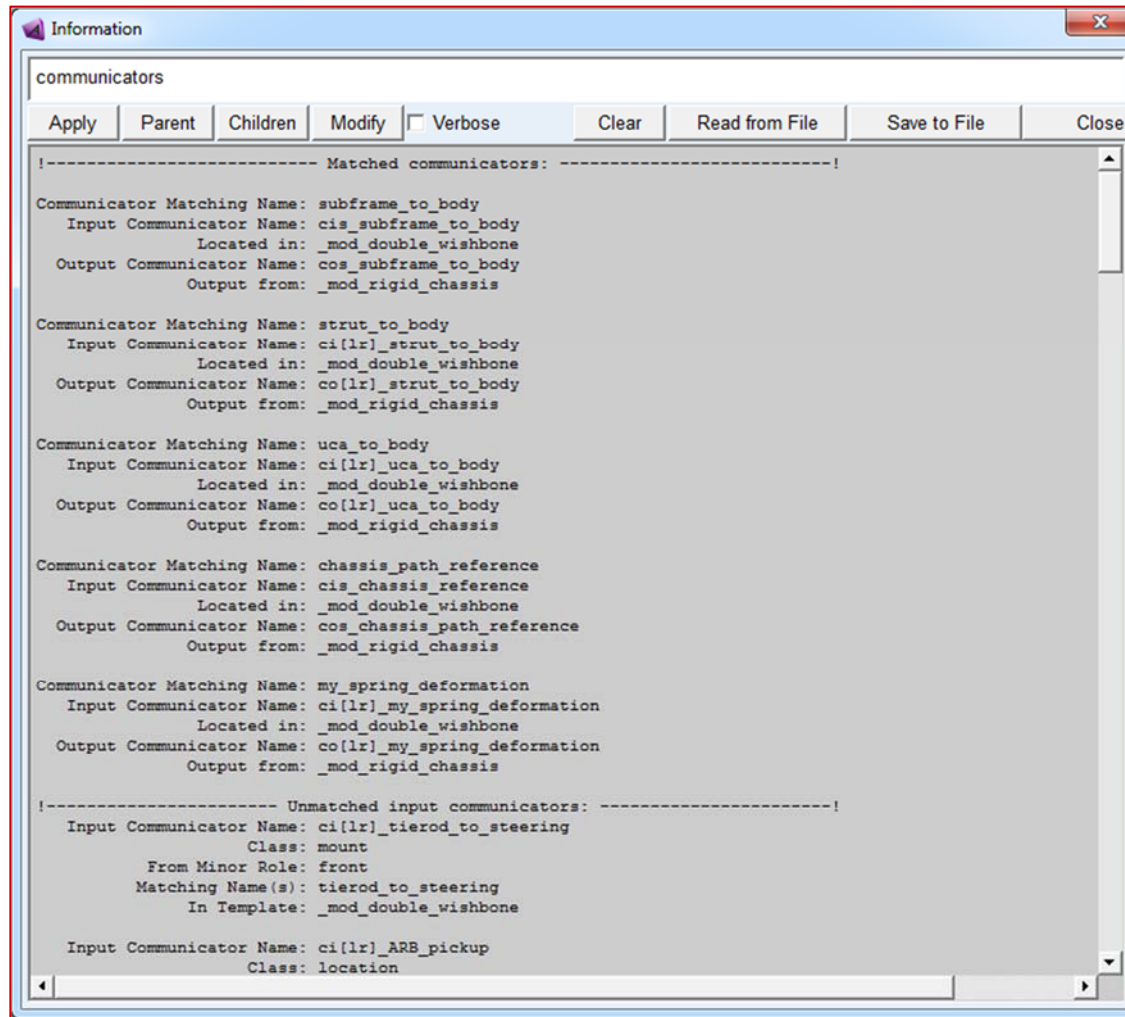
1. Go to **Build** menu and select **Communicator**, point to test
2. Right click the **test** dialog box to select the two templates
3. Enter the Minor roles



Requests

- Communicator test (Cont.):

4. Press **OK** on the dialog box to display the information.



Requests

- **Create subsystem from the templates:**
 1. Go to **Tools** and select **Standard Interface**
 2. Create new subsystems **mod_TR_Front_suspension** and **mod_body** for the **double_wishbone** and **rigid_chassis** models.
 3. For **mod_TR_Front_suspension**, go to **Adjust >> Driveline Activity...** and select **Inactive**
 4. Save the subsystems in **acar_training** database.

Requests

- **Replacing the front suspension and body subsystems from the demo vehicle with new subsystems:**
 1. Open the **MDI_demo_vehicle** from the **acar_shared** database.
 2. Go to **File** menu and select **Manage Assemblies**, point to **Replace Subsystems**.
 3. Replace the original body and front suspension subsystems, first with **mod_body.sub**, then **mod_TR_Front_suspension.sub**

Requests

- **Run analysis and compare results**
 1. Run a simple ISO lane change simulation :
 2. Select **Simulate → Full –Vehicle Analysis → Course Events → ISO Lane Change**.
 3. After the analysis, your request results can be viewed in PostProcessor by selecting Requests in the Sources drop-down menu then expanding **+mod_TR_Front_suspension** in the Request field.

WORKSHOP 15

EXPLORING AND COMPLETING TEMPLATES

Exploring and Completing Templates

- **Problem Statement:**

- This workshop tests some of the topics about template building, including defining communicators and joints, and investigating your template with the database navigator.

- **We recommend that whenever possible, you modify existing templates, rather than create new ones. To be able to modify existing templates and to customize them for your use, you must be able to understand them very well.**



This workshop takes about one and a half to two hours to complete.

Exploring and Completing Templates

- **Defining your template**

To define your template:

1. Copy **_steer_training.tpl** (available in **acar_training_MD** database) to the **template.tbl** directory of your choice (for example, place it in **acar_training.cdb/templates.tbl**).
2. Open **_steer_training.tpl** in template-builder mode.
A rack and pinion steering template appears. The template is incomplete: it needs joints to be defined, as well as communicators and mount parts. Use the Database Navigator to investigate the template by looking at the parts and icons, and try to determine what yet needs to be defined for this template.
3. Make the necessary changes to define your template properly. To initiate exploration of your model and challenge yourself to determine what entities still need to be defined, first see General steps to define your template, on the next slide. If you have trouble or would like to check your work, see Detailed steps to define your template, on slide 7 to determine what changes should be made.

Exploring and Completing Templates

- **General steps and tasks to define your template**

To define your template:

1. Constrain the motion of the steering columns to each other.
2. Constrain the steering wheel joint to the steering column joint so that one turn of the steering wheel causes one turn of the steering column. (Hint: Use a gear to do this. To learn more about gears, select the dialog box, and then press **F1**.)
3. Constrain the motion of the steering shaft to the rack housing.
4. Constrain the motion of the rack to the rack housing.
5. Constrain the motion of the steering shaft to the rack (Hint: Use a gear to do this).
6. Create a mount part for the rack housing; name it **rackhousing_to_body**.

Exploring and Completing Templates

7. Make sure the rack will be able to connect to the MacPherson template you created earlier. If necessary, create any mount parts or communicators.
8. Check that the steering column housing will mount properly. If necessary, create any mount parts or communicators.
9. Create an output communicator named **steering_rack_joint**, which provides the name of the rack joint so the built-in Adams/Car test rigs can apply a motion there.
10. If time allows, add location communicators to your template to locate the ends of the rack to the ends of the tie rod.

Exploring and Completing Templates

- **Detailed steps and tasks to define your template**

To define your template:

1. Create two hooke (universal) joints: one between the steering column and the intermediate shaft, and one between the intermediate shaft and the steering shaft.
2. Create a reduction gear to constrain the revolute joint for the steering wheel to the cylindrical joint of the steering column. The reduction gear should always be active. Use a ratio of 1, so that one turn of the steering wheel causes one turn of the steering column. The `__MDI_SDI_TESTRIG` expects a revolute joint, and is the reason why this modeling is used in the steering template.
3. Create a revolute joint between the end of the steering shaft and the rack housing.
4. Create a translational joint between the rack and the rack housing.

Exploring and Completing Templates

5. Create a reduction gear that constrains the rotational motion of the steering shaft to the translational motion of the rack (use a reduction ratio of 0.05).
6. Create a mount part that will attach to the body; name it **rackhousing_to_body**, place it at **cfs_rack_mount** (to create a single mount part), and attach it to the rack housing with a fixed joint so that the rack housing will be fixed to the body.
7. Open the MacPherson template created earlier, and check the mount parts and mount communicators at the tie rods. The mount input communicator **tierod_to_steering** in the MacPherson template requires that a mount output communicator named **tierod_to_steering**, which outputs the rack part, be defined in the template *steer_training*. Create a left/right mount output communicator that outputs the *single* part **ges_rack**.
8. The steering column is already attached to a mount part named **steering_column_to_body**.

You will edit a body template in Workshop 16—Full-Vehicle Assembly to make sure these communicators match properly.

Exploring and Completing Templates

9. Create a single entity type **joint_for_motion** output communicator named **steering_rack_joint**, which outputs the rack joint you created in Step 4, on slide 7. The communicator will provide the joint name so the built-in Adams/Car test rigs can apply a motion there.



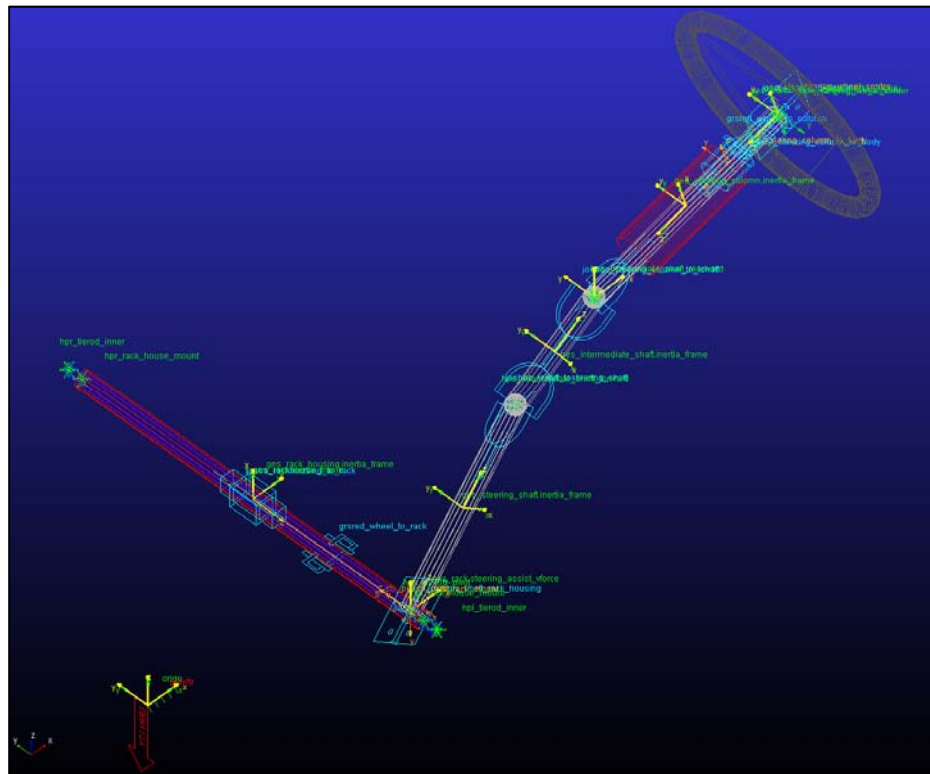
For details on how this communicator is used in the test rigs, see the Templates section in the Adams/Car online help or investigate the test rigs themselves.

Your template should look like `steer_final.tpl`, provided in the **acar_training_MD** database and will be used in the final full-vehicle workshop.

Exploring and Completing Templates

10. If time allows, add location communicators to your template to locate the ends of the rack to the ends of the tie rod. See SimCompanion article KB8014797 at:

<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8014797>



Exploring and Completing Templates

- The following lists all of the communicators you should have before and after this workshop (to get this information, select *Build* → *Communicator* → *Info*):
- Before:

```
Listing of input communicators in '_steer_training'
-----

Communicator Name:      Entity Class:      From Minor Role:      Matching Name:
cis_steering_column_to_body      mount      inherit      steering_column_to_body

1 input communicator was found in '_steer_training'

Listing of output communicators in '_steer_training'
-----

Communicator Name:      Entity Class:      To Minor Role:      Matching Name:
cos_max_rack_displacement      parameter_real      inherit      max_rack_displacement
cos_max_rack_force      parameter_real      inherit      max_rack_force
cos_max_steering_angle      parameter_real      inherit      max_steering_angle
cos_max_steering_torque      parameter_real      inherit      max_steering_torque
cos_steering_wheel_joint      joint_for_motion      inherit      steering_wheel_joint

5 output communicators were found in '_steer_training'
```

Exploring and Completing Templates

- After:

```
Listing of input communicators in '_steer_training'
-----

Communicator Name:          Entity Class:      From Minor Role:    Matching Name:
cis_rackhousing_to_body    mount           inherit            rackhousing_to_body
cis_steering_column_to_body mount           inherit            steering_column_to_body

2 input communicators were found in '_steer_training'

Listing of output communicators in '_steer_training'
-----

Communicator Name:          Entity Class:      To Minor Role:      Matching Name:
co[lr]_tierod_to_steering  mount             inherit             tierod_to_steering
cos_max_rack_displacement  parameter_real    inherit             max_rack_displacement
cos_max_rack_force         parameter_real    inherit             max_rack_force
cos_max_steering_angle     parameter_real    inherit             max_steering_angle
cos_max_steering_torque    parameter_real    inherit             max_steering_torque
cos_steering_rack_joint    joint_for_motion  inherit             steering_rack_joint
cos_steering_wheel_joint   joint_for_motion  inherit             steering_wheel_joint

8 output communicators were found in '_steer_training'
```

Exploring and Completing Templates

- **Tips for exploring templates**

Note: Because you perform many steps in this section in the Database Navigator, make sure you have the Database Navigator displayed (**Tools → Database Navigator**).

- **Investigating model topology**

- **To list parts and connections:**

- To see parts and connections, set the option menu at the bottom of the Database Navigator to **Bodies**, **Constraints**, or **Forces**.
- Set the option menu at the top to the type of information you want to see.

Exploring and Completing Templates

- **To view the topology of parts:**
 1. From the option menu at the top of the dialog box, select **Topology by Parts** or **Topology by Connections**.
 2. From the tree list or view window, select an object.
- **The topology of the object appears in the text box to the right.**
- **To graphically view the topology of parts:**
 1. From the option menu at the top of the dialog box, select **Graphical Topology**.
 2. From the tree list or view window, select an object.

A graphical display of the object's topology appears in the text box to the right.

Exploring and Completing Templates

- **Viewing the associativity of objects**

- You can use the Database Navigator to display the objects that a selected object uses. For example, you can select a joint in the tree list to show the I and J markers that the joint uses. You can also select to view the objects that use the selected object.

- **To view the associativity of objects:**

1. From the option menu at the top of the dialog box, select **Associativity**.
2. Set the associativity:
 - To show the objects that the selected object uses, select **Uses**.
 - To show the objects that use the selected object, select **Is Used By**.
3. From the tree list or view window, select an object.

The objects associated with the selected object appear in the text box to the right.

Exploring and Completing Templates

- **To set up automatic navigation of the objects:**
 - To see what objects are dependent on a certain object, select **Auto Navigate**.
- **To save the current associativity information to a file:**
 - Select **Save to File**.
- **Viewing object information through the Database Navigator**
 - You can use the Database Navigator just as you would the Information window to display information about the selected object.
- **To display object information:**
 1. Set the option menu at the top of the Database Navigator to **Information**.
 2. From the tree list or view window, select an object.
 3. The information about the object appears in the text box to the right.

Exploring and Completing Templates

- **To save the information to a file:**
 - Select **Save to File**.
- **To list the type of communicators:**
 1. From the **Build** menu, point to **Communicator**, and then select **Info**.
 2. Set **Type** to **Input**.
 3. Set **Entity** to **All**.
 4. Select **OK**.
- **To see what is dependent on a communicator:**
 1. From the **Tools** menu, select **Database Navigator**.
 2. In the **Filter** text box, enter **ci***.
 3. Set the option menu at the top of the dialog box to **Associativity**.
 4. To get the relevant information, select **Uses** and **Is Used By**.

Exploring and Completing Templates

- **Location and orientation parametrics:**

- To find out how the parametric dependencies are set up, you can regard two entities as starting points: hardpoints and input communicators.
- Hardpoints have no dependencies. Therefore, you can regard them as starting points for parametrics. You can parameterize construction frames and other components to hardpoints.
- You use parameter variables to parameterize the location and orientation of construction frames.



For more information, see the Adams/View online help.

Exploring and Completing Templates

- **To see what is dependent on a hardpoint:**

1. In the **Filter** text box, enter **hp***.
2. Select a hardpoint.
3. Select **Apply**.

The Information window appears, displaying hardpoint location values.

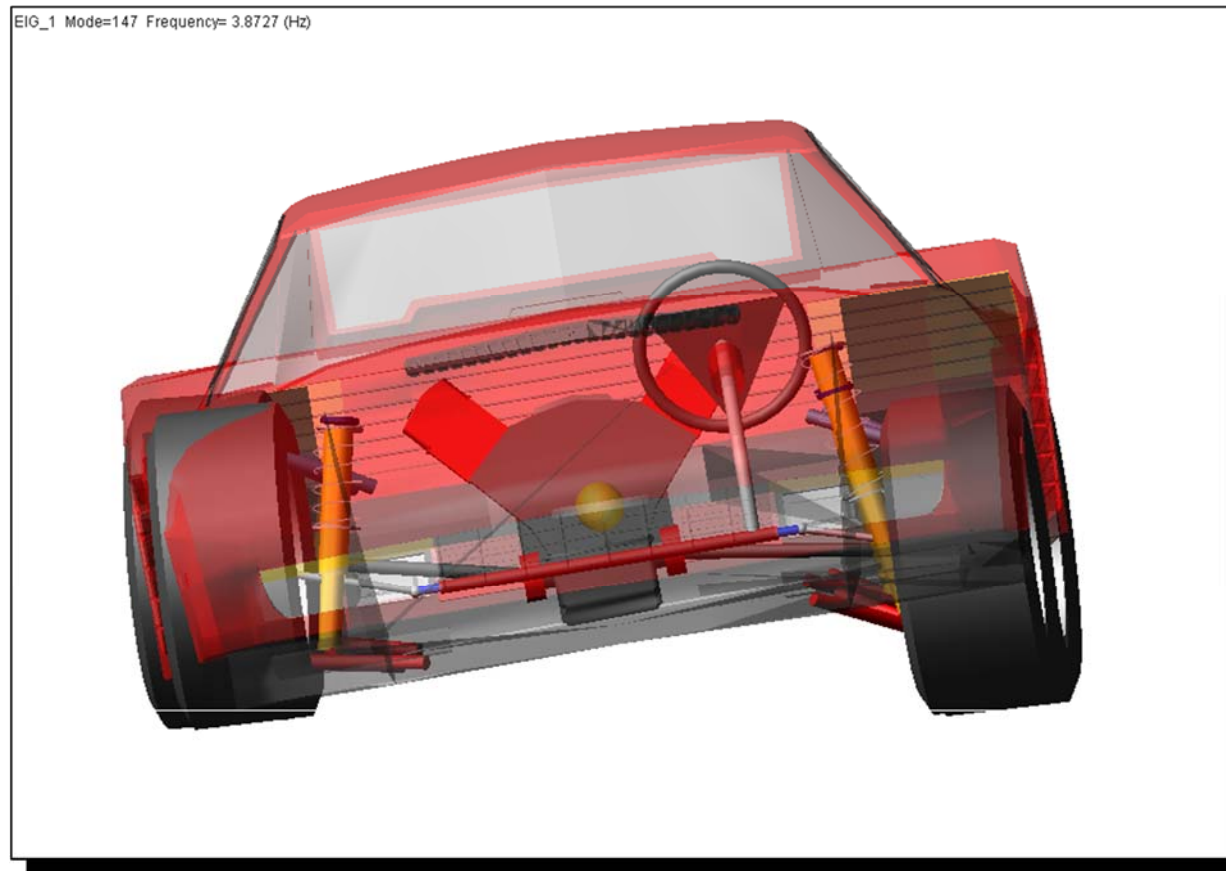
4. In the Database Navigator, set the option menu at the top of the dialog box to **Associativity** to see a list of dependents. Also, you can select Highlight to see the hardpoint position.
5. You can right-click each construction frame and select **Info** or **Modify**. The information displayed includes information for the parametric locations in the template. For construction frames, you look both at the expression to see how the construction frame is parametric to other construction frames and hardpoints, and what is dependent on the construction frame.

Exploring and Completing Templates

- **To see what is dependent on construction frames:**
 1. In the **Filter** text box, enter **cf***.
 2. Set the option menu at the top of the dialog box to **Associativity**.
- **To list dependencies for parameter variables:**
 1. In the **Filter** text box, enter **pv***.
 2. To see the parameter variables, set the option menu at the top of the dialog box to **Information**.
 3. To see a list of dependents, set the option menu at the top of the dialog box to **Associativity**.

WORKSHOP 16

LINEAR MODES ANALYSIS



Linear Modes Analysis

- **Problem statement**

- In this workshop you will compute the linear modes of a full-vehicle at static equilibrium.
- The linear modes of the virtual system will be animated and compared to the expected results of the physical system.
- You will also run a step steer full-vehicle analysis and perform linear analysis at the end of the simulation



This workshop takes about 30 minutes to complete

Linear Modes Analysis

- **Getting Started:**
 - Load the Adams/Car Demo Vehicle (**MDI_Demo_Vehicle**) from the **acar_shared** database.
- **To Run a Linear Modes Analysis in Adams/Car:**
 - Linear analyses are done about some system operating point, typically a static configuration of the vehicle. To solve for static equilibrium and then perform a linear analysis:
 - **Simulate → Full Vehicle Analysis → Static and Quasi-Static Maneuvers → Static Equilibrium...**
 - Enter **linear_modes** as **output prefix**.
 - Specify a **Static Set-Up** of type **Settle**
 - Select the **Perform Linear** checkbox and specify **Damping**
 - Hit **OK** or **Apply** to run the static, then linear, analyses.

Linear Modes Analysis

- **To Review and Animate the System Modes:**
 - Linear modes can be viewed in either tabular form or as animations. To view the mode shape information in tabular form:
 - Switch to Adams/PostProcessor by hitting the **F8** key
 - Right click in the main PostProcessor window and select **Load Mode Shape Animation**
 - Next, click the **Table of EigenValues** button in the dashboard near the bottom. In the table of eigen values note how the Mode Number, **Undamped Natural Frequency** and **Damping Ratio** are reported as columns.
 - Scroll down through the **Damping Ratio** column and note how the first 144 modes are mostly over-damped, having a Damping Ratio of 1.0, or 100%. Find the number of the first mode to have a damping ratio less than 1.0. This mode should be around #145 and should have a damping ratio of about 0.27, or 27%.
 - Enter this mode number (likely #145) in the **Mode Number** field in the Adams/PostProcessor Dashboard. Hit the Animate button to view an animation of this mode shape.

Linear Modes Analysis

- **Animate More System Modes:**
 - Scroll through the modes above the previous one. Identify some common vehicle characteristics by animating the modes and then comparing with the chart below:

Mode #	Description	Frequency (Hz)	Damping (%)
146	Body Pitch	3.0	46.0
147	Body Roll	4.4	46.7
145	Body Bounce	1.5	27.4
152	Rear Wheel Hop	19.0	14.6

Linear Modes Analysis

- Here we will run a Step Steer analysis and perform a linear analysis at the end of that simulation
- To Run a Step Steer event:
 1. Simulate → Full Vehicle Analysis → Open-Loop Steering Events → Step Steer...
 2. Fill in the dialog box as shown on the right:
 3. Click **OK**.

Full-Vehicle Analysis: Step Steer

Full-Vehicle Assembly	MDI_Demo_Vehicle
Assembly Variant	default
Output Prefix	linear
End Time	5
Number Of Steps	50
Simulation Mode	files_only
Road Data File	mdids://acar_shared/roads.tbl/2d_flat.rdf
Initial Velocity	30 km/hr
Gear Position	5
Final Steer Value	90
Step Start Time	0
Duration of Step	3
Steering Input	Angle

☐ Cruise Control

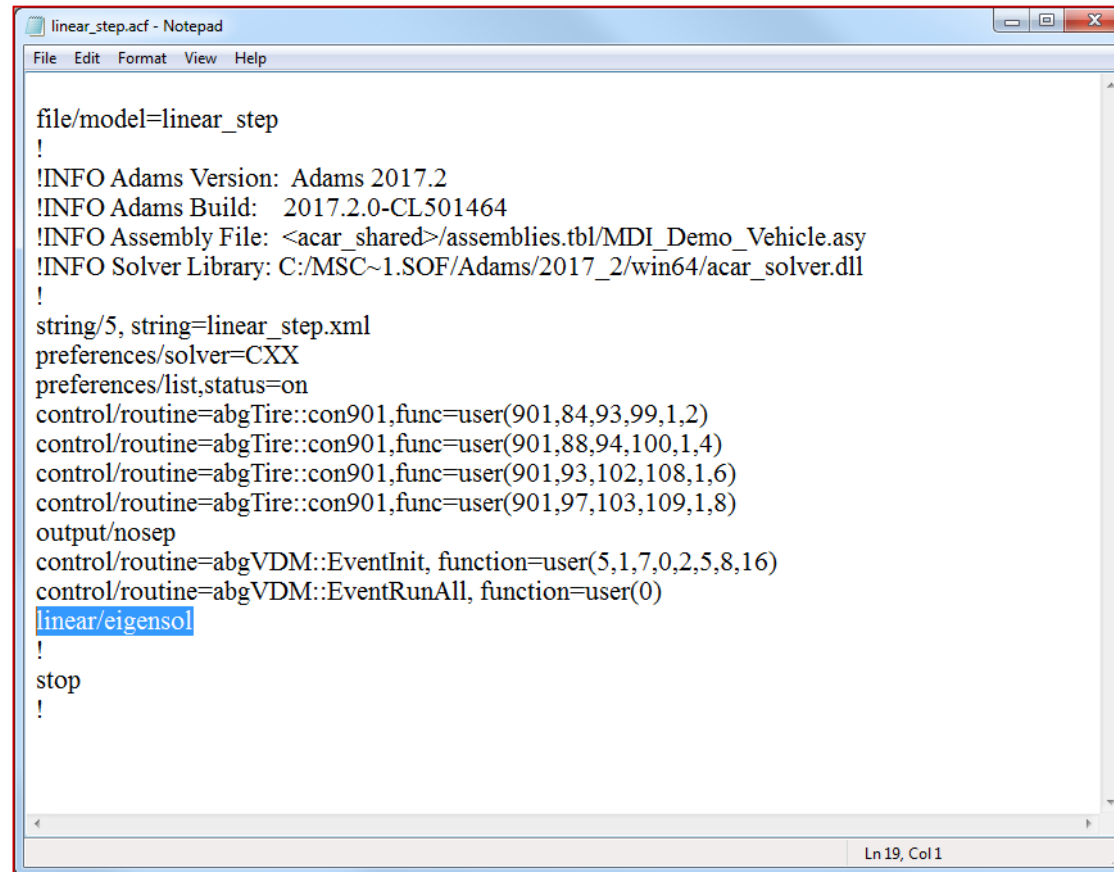
☒ Quasi-Static Straight-Line Setup

☒ Create Analysis Log File

OK Apply Cancel

Linear Modes Analysis

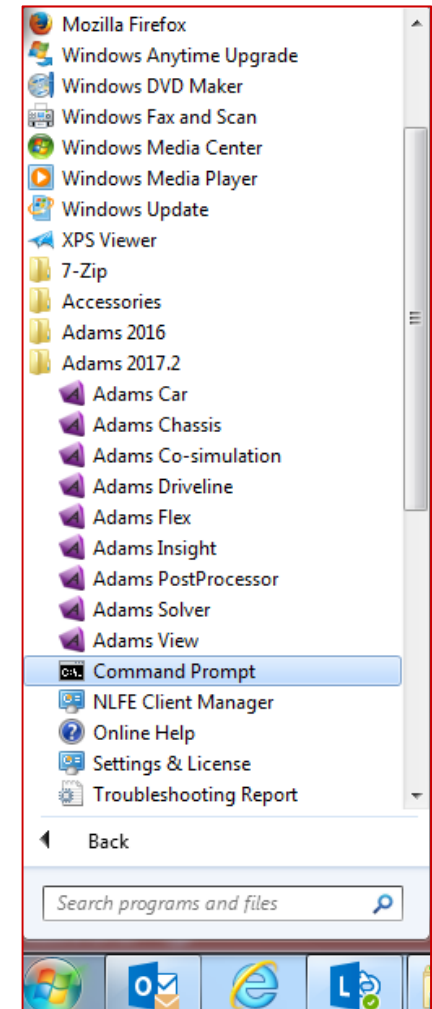
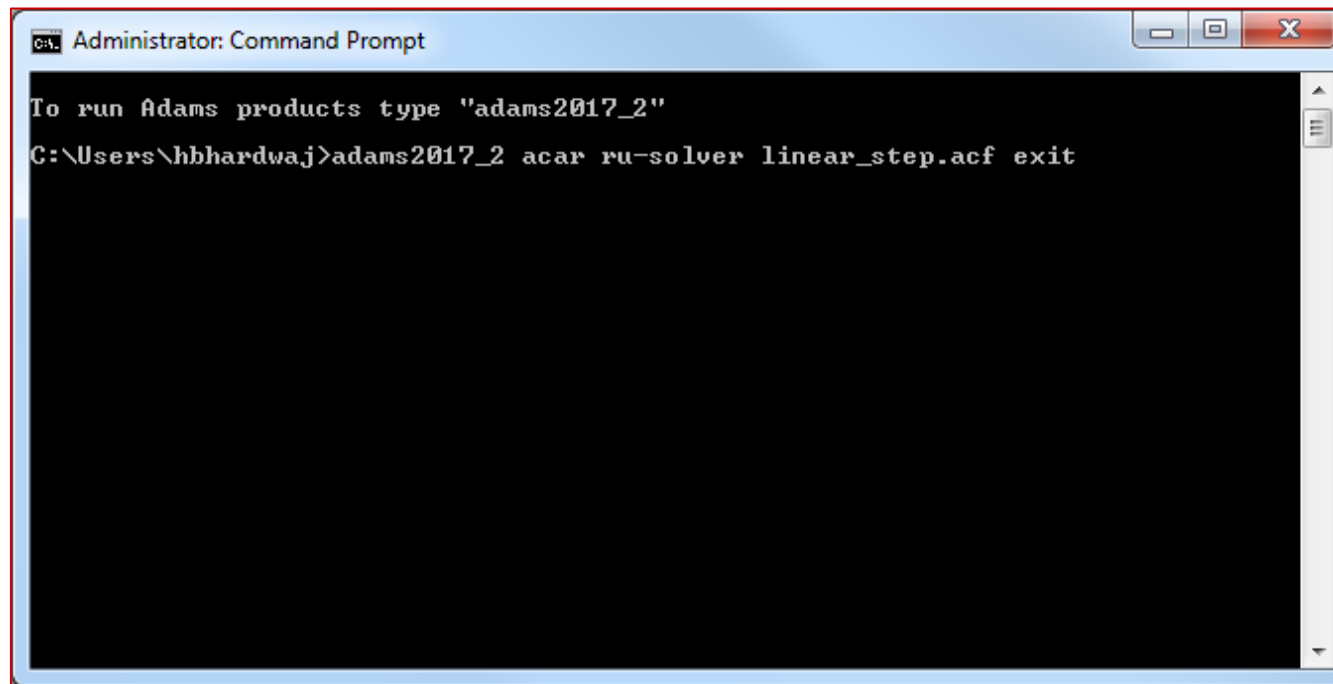
- **Modify the acf file to include the linear/eigensol statement**
 - Find the **linear_step.acf** in your working directory, open it using any text editor.
 - Add a line: “**linear/eigensol**” above the “**stop**” command
 - **Save the file**



```
file/model=linear_step
!
!INFO Adams Version: Adams 2017.2
!INFO Adams Build: 2017.2.0-CL501464
!INFO Assembly File: <acar_shared>/assemblies.tbl/MDI_Demo_Vehicle.asy
!INFO Solver Library: C:/MSC~1.SOF/Adams/2017_2/win64/acar_solver.dll
!
string/5, string=linear_step.xml
preferences/solver=CXX
preferences/list,status=on
control/routine=abgTire::con901,func=user(901,84,93,99,1,2)
control/routine=abgTire::con901,func=user(901,88,94,100,1,4)
control/routine=abgTire::con901,func=user(901,93,102,108,1,6)
control/routine=abgTire::con901,func=user(901,97,103,109,1,8)
output/nosep
control/routine=abgVDM::EventInit, function=user(5,1,7,0,2,5,8,16)
control/routine=abgVDM::EventRunAll, function=user(0)
linear/eigensol
!
stop
!
```

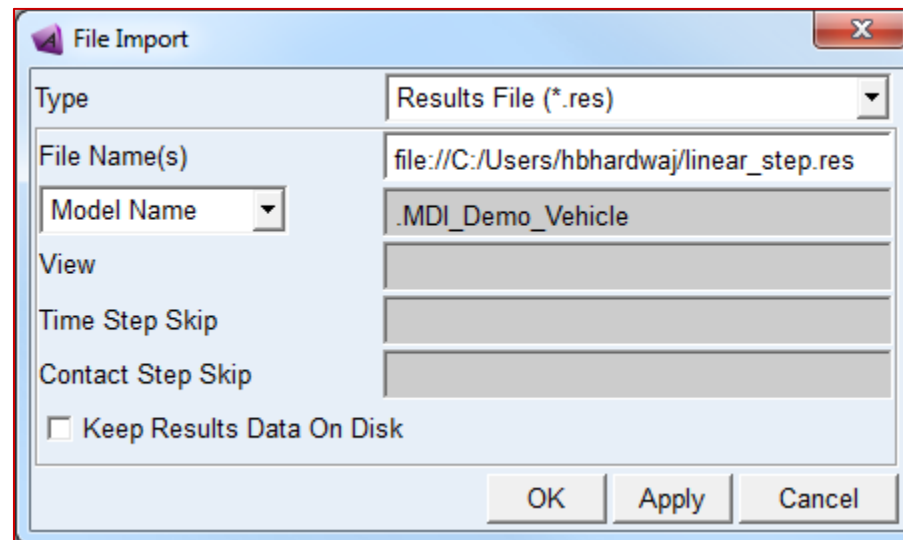
Linear Modes Analysis

- Run the script using Adams/Car Solver
 - Open **Adams - Command Prompt** (start → adams 2017.2 folder → command Prompt)
 - Issue the command:
adams2017_2 acar ru-solver linear_step.acf exit



Linear Modes Analysis

- **Load the simulation result**
 - In Post Processor, **File** → **Import** → **Results File**
 - Load the **linear_step.res** in your working directory to the **MDI_Demo_Vehicle** model

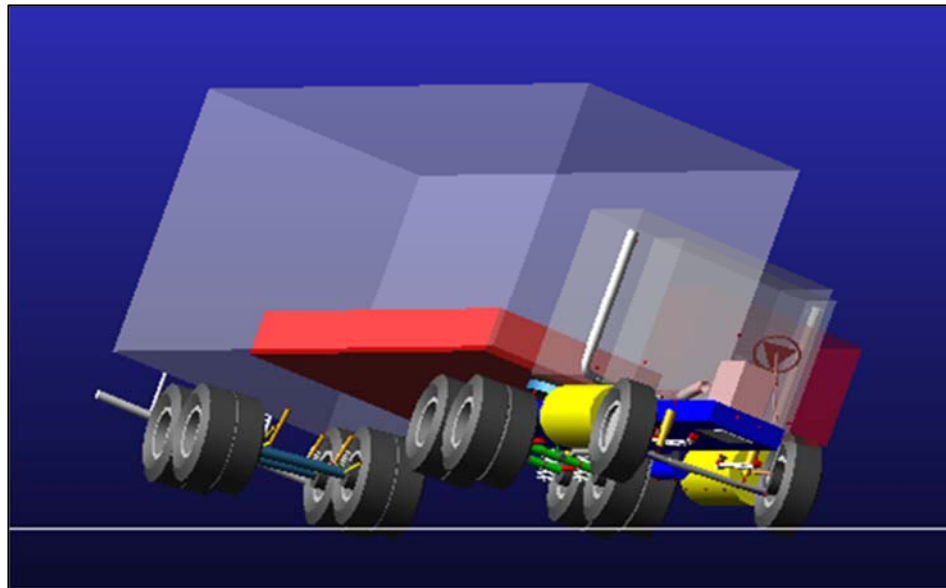


Linear Modes Analysis

- **Review the mode shape animation**
 - Load the mode shape animation as you did previously and notice the vehicle is in a different configuration. Compare the mode shapes with the previous static equilibrium run.

WORKSHOP 17

TILT TABLE ANALYSIS



Tilt Table Analysis

- **Problem Statement:**

- Investigate Tractor Semi-Trailer roll-over behavior using both tilt-table and full-vehicle simulations.



This workshop takes one hour to complete.

Tilt-Table Analysis on the Semi-Trailer Assembly

- To run the Tilt-Table Analysis on Semi-Trailer truck:

1. Start the Adams/Car in Standard Interface.

2. Load the Truck Plugin:

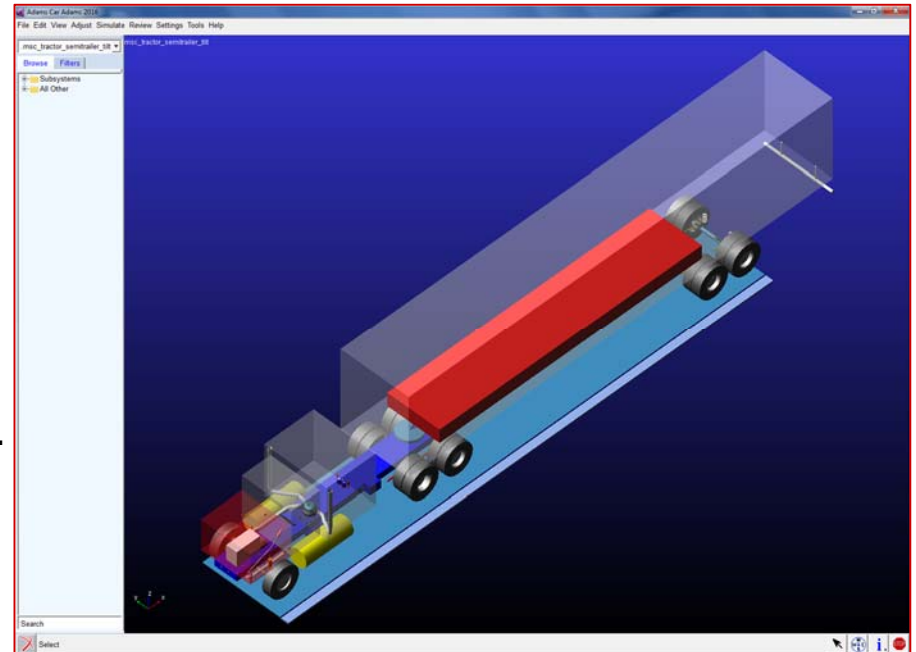
- a. Select **Tools > Plugin Manager**

- b. Check Load for Adams/Car Truck

- c. Select **Apply**.

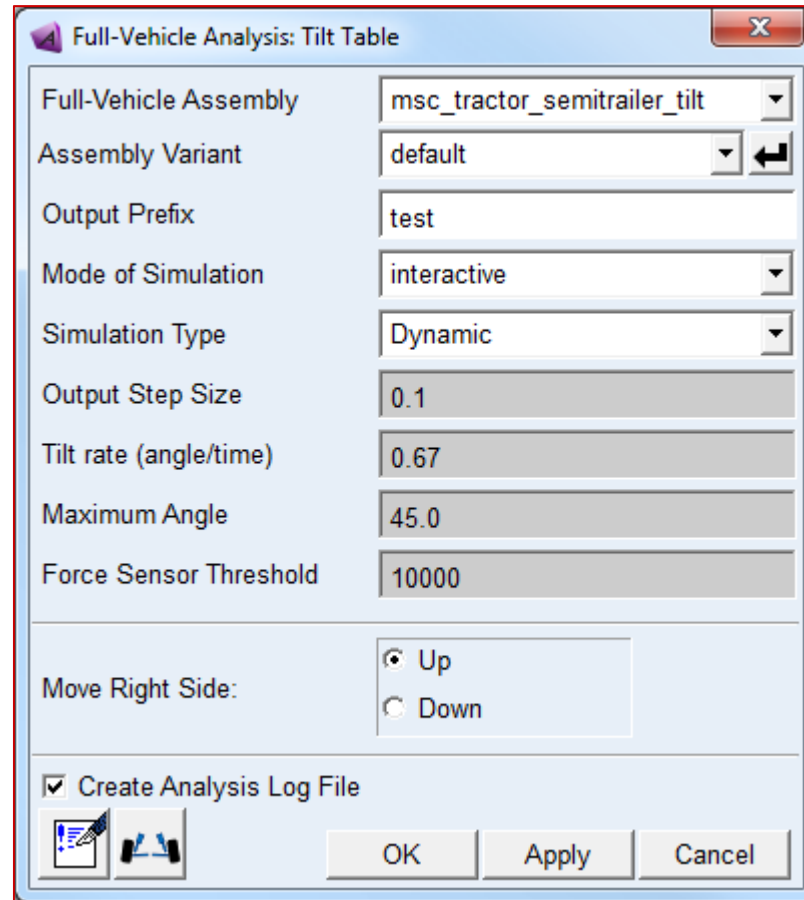
3. Open the assembly:

Go to **File – Open – Assembly**,
right click and select
msc_tractor_semitrailer_tilt.asy
from the **atruck_shared** database.



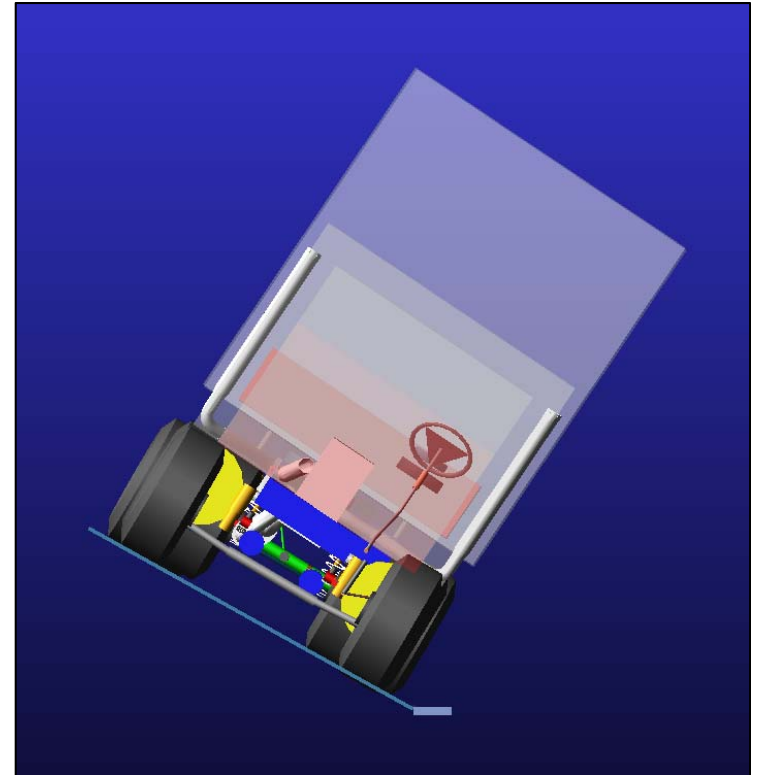
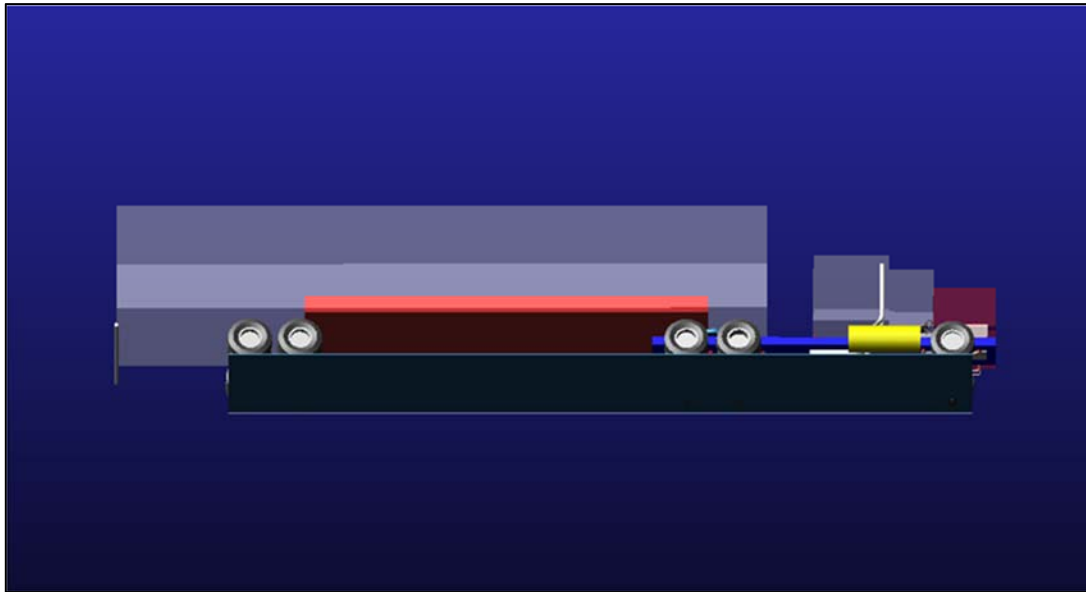
Run a Tilt Table Analysis

4. Select **Simulate > Full Vehicle Analysis > Roll Stability Events > Tilt Table Analysis**
5. Fill out the dialog box, setting the Force Sensor Threshold as shown:
6. Select **Apply**.



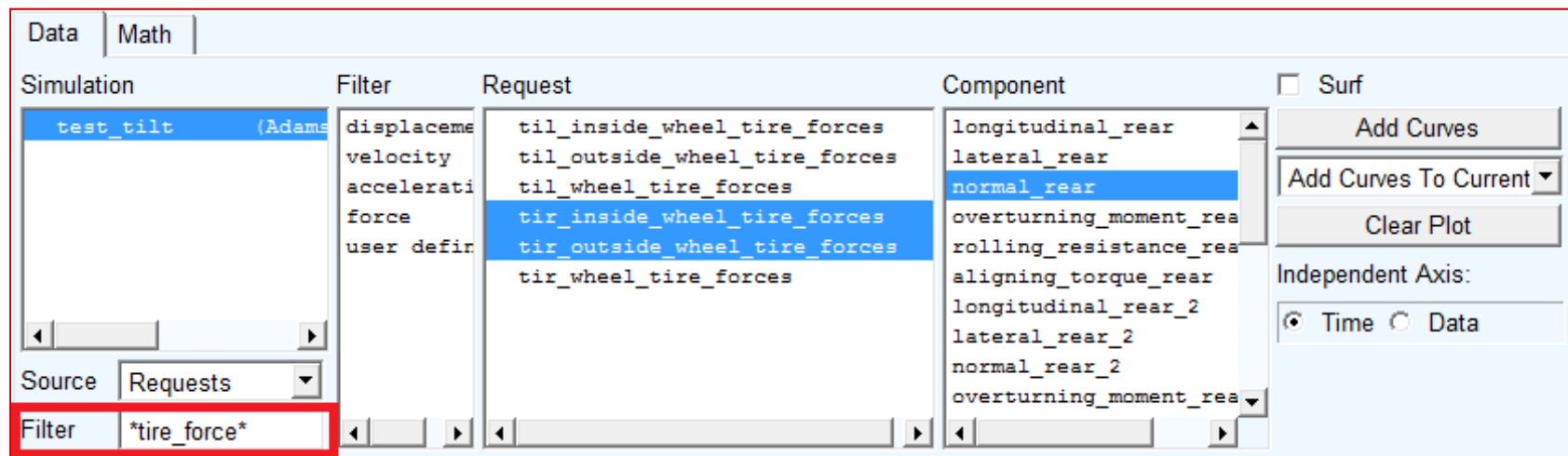
Run a Tilt Table Analysis

7. When the simulation completes, to view the animation, select **Review > Animation Controls**



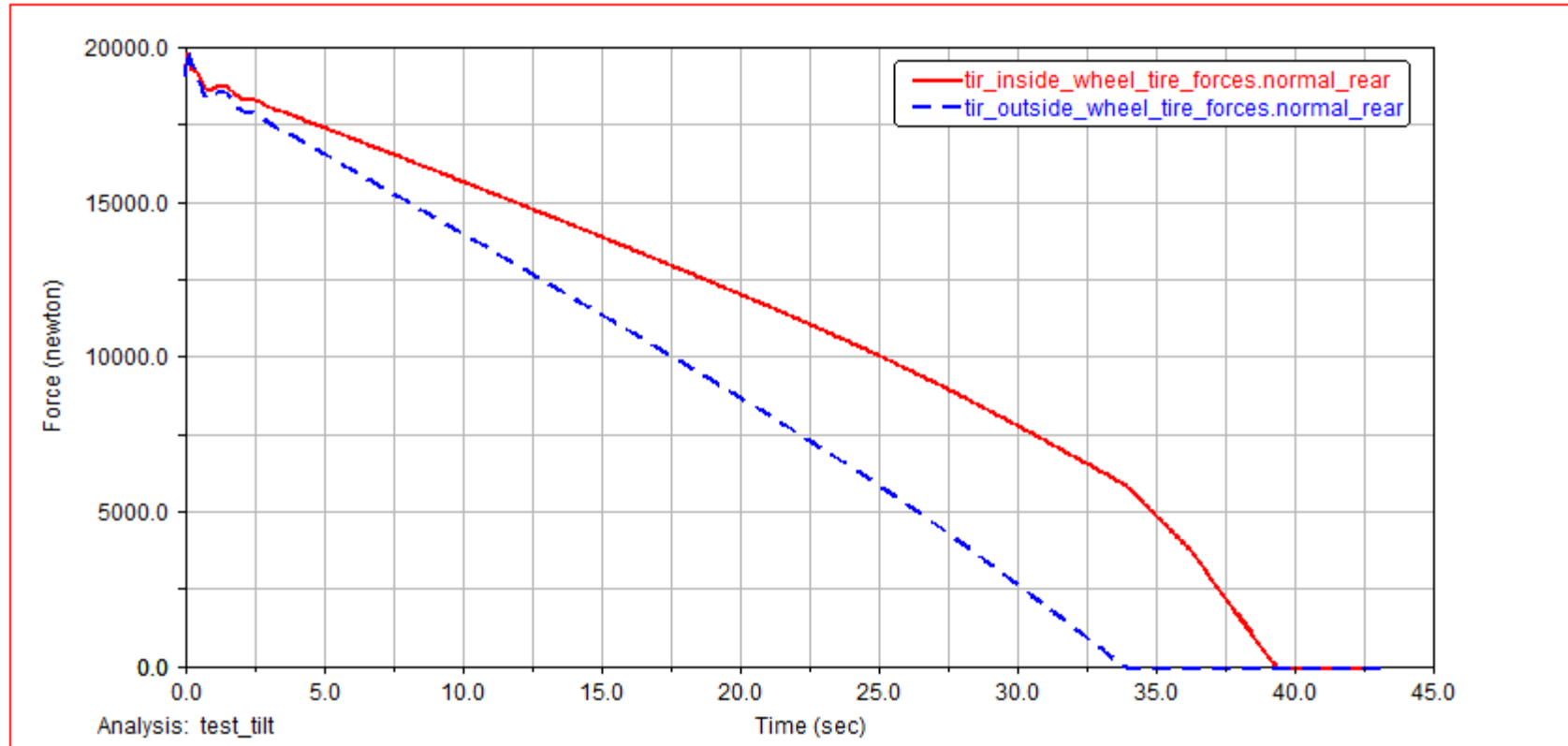
Run a Tilt Table Analysis

8. Switch to Adams/Postprocessor window (Press F8) to view the result of simulation, type ***tire_force*** into the Filter field
9. Select both **tir_inside_wheel_tire_forces** and **tir_outside_wheel_tire_forces** under Request and **normal_rear** under Component menu.
10. Select **Add Curves**.



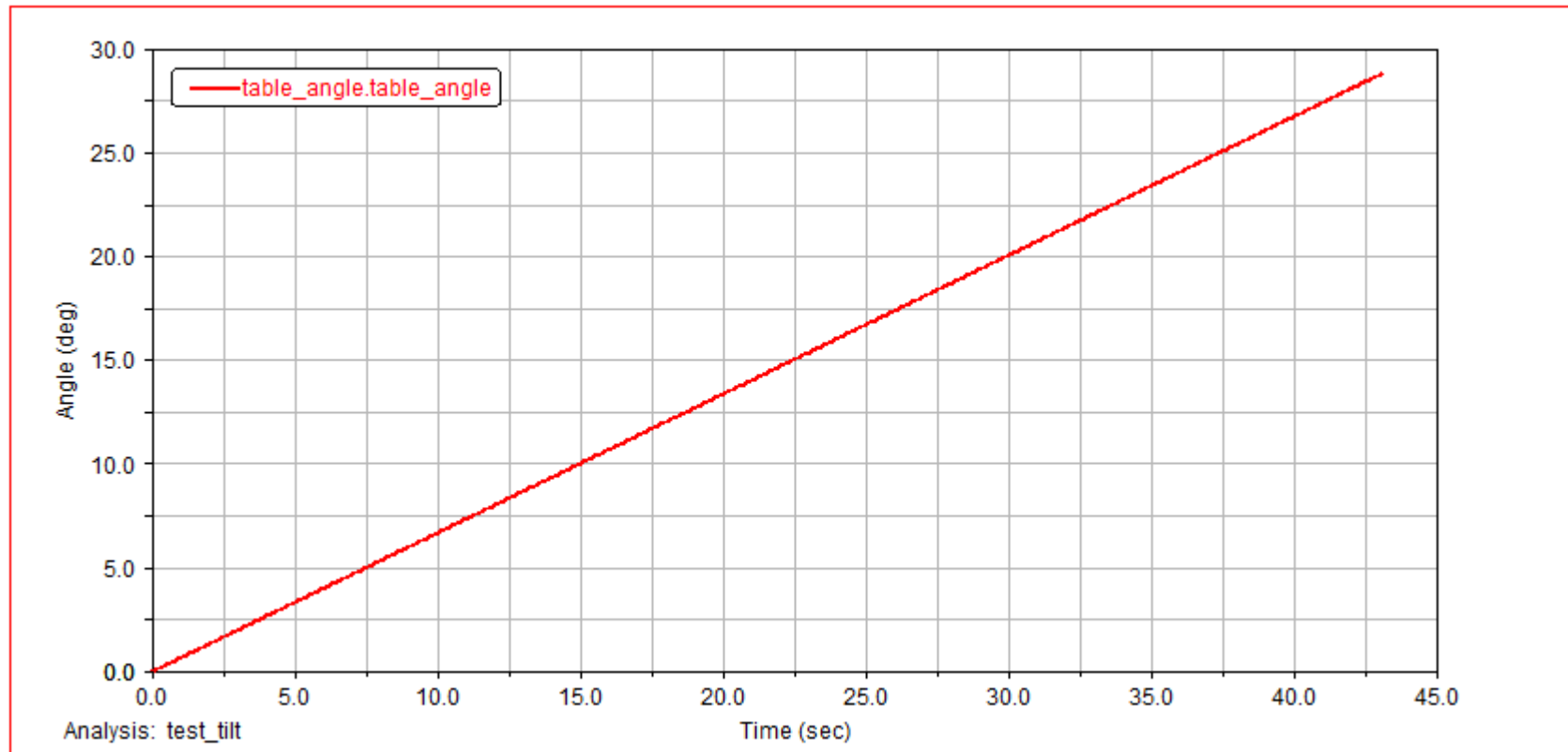
Run a Tilt Table Analysis

11. This produces output like the following:



Run a Tilt Table Analysis

12. Now replace the text in the Filter field with ***angle*** and select **table_angle** under requests to observe the table tilt off:



Run a Single Lane Change Analysis

- To Run a Single Lane Change maneuver with the model:
 1. Open the assembly **msc_tractor_semitrailer.asy** from the **atruck_shared** database.
 2. Select **Simulate > Full Vehicle Analysis > Open-Loop Steering > Single Lane Change**.
 3. Fill out the **Single Lane Change** dialog box as shown.

Note: Select **2d_flat_high_mu.rdf** road from the **acar_training_MD** database.

Full-Vehicle Analysis: Single Lane Change

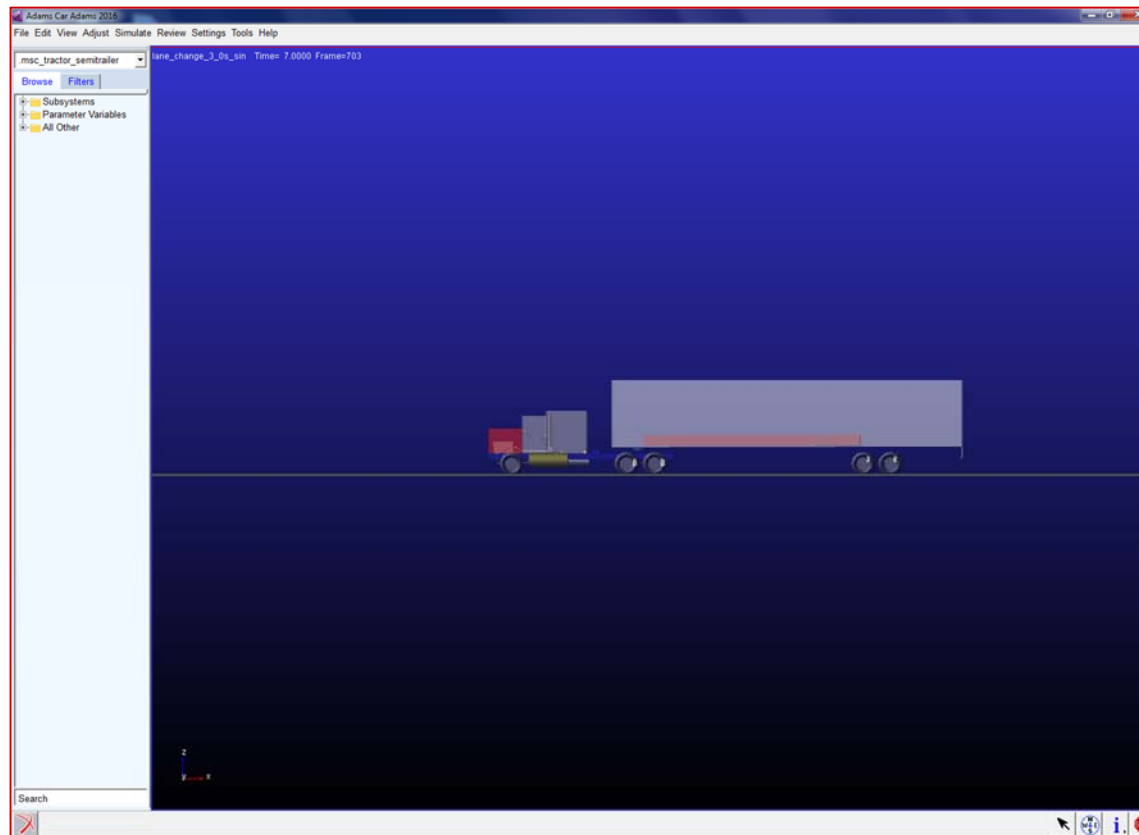
Full-Vehicle Assembly	msc_tractor_semitrailer	
Assembly Variant	default	
Output Prefix	lane_change_3_0s	
End Time	7	
Number Of Steps	700	
Simulation Mode	interactive	
Road Data File	acar_training_MD.cdbroads.tbl\2d_flat_high_mu.rdf	
Initial Velocity	70	km/hr
Gear Position	5	
Maximum Steer Value	360	
Start Time	1	
Cycle Length	3	
Steering Input	Angle	
<input checked="" type="checkbox"/> Cruise Control		
<input checked="" type="checkbox"/> Quasi-Static Straight-Line Setup		
<input checked="" type="checkbox"/> Create Analysis Log File		

OK Apply Cancel

Run a Single Lane Change Analysis

4. Select **Apply**
5. When the simulation is finished, select **Review > Animation Controls**, then play the animation

The truck makes wheel lift off, but recovers and does not roll.



Run a Single Lane Change Analysis

- Re-run with a slower steer execution time:
 1. Again select **Simulate > Full Vehicle Analysis > Open-Loop Steering > Single Lane Change**
 2. Use the modified values shown:
 3. Hit **Apply** to run the simulation.

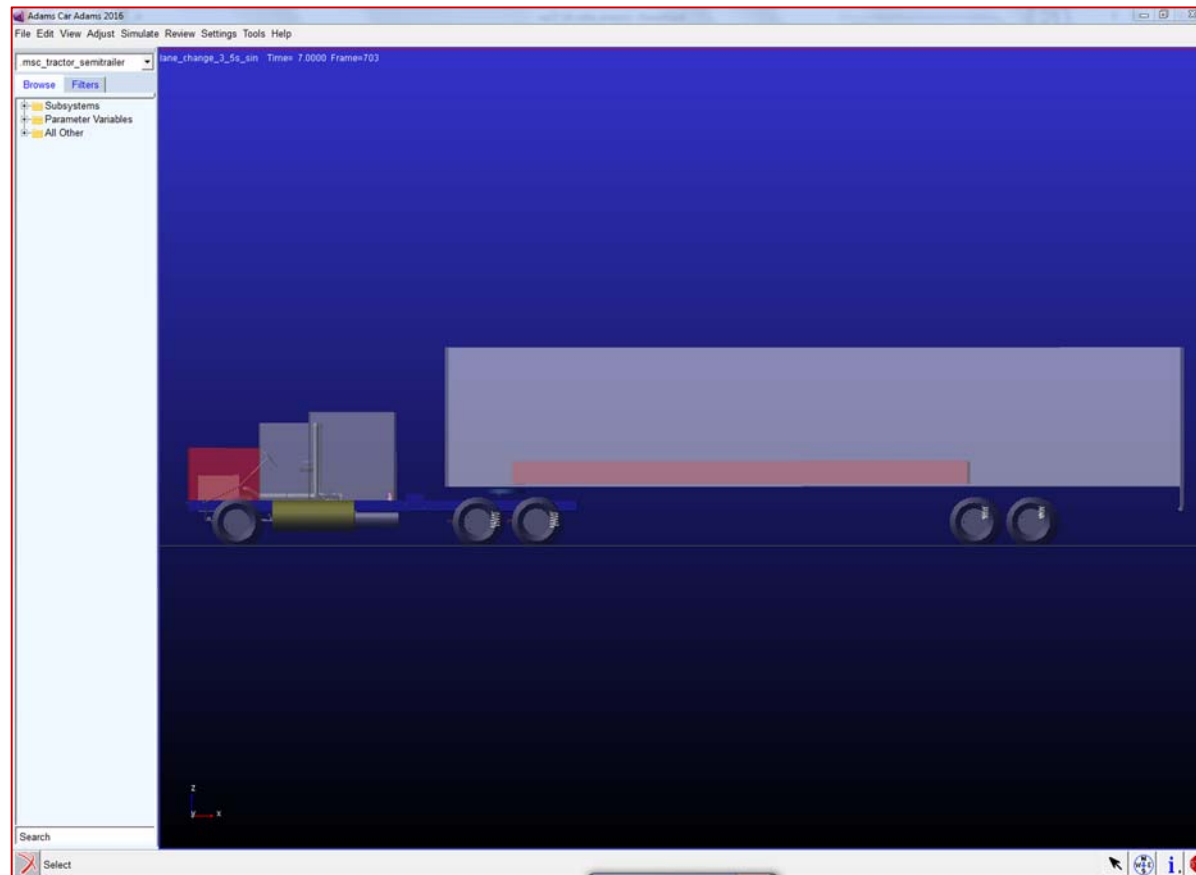
Full-Vehicle Analysis: Single Lane Change

Full-Vehicle Assembly	msc_tractor_semitrailer
Assembly Variant	default
Output Prefix	lane_change_3_5s
End Time	7
Number Of Steps	700
Simulation Mode	interactive
Road Data File	acar_training_MD.cdb\roads.tb\l2d_flat_high_mu.rdf
Initial Velocity	70 km/hr
Gear Position	5
Maximum Steer Value	360
Start Time	1
Cycle Length	3.5
Steering Input	Angle
<input checked="" type="checkbox"/> Cruise Control	
<input checked="" type="checkbox"/> Quasi-Static Straight-Line Setup	
<input checked="" type="checkbox"/> Create Analysis Log File	

OK Apply Cancel

Run a Single Lane Change Analysis

- For this analysis the truck rolls over, perhaps because the cycle length of 3.5 seconds results in an input stimulation that is closer to the effective roll frequency of the truck during the maneuver.



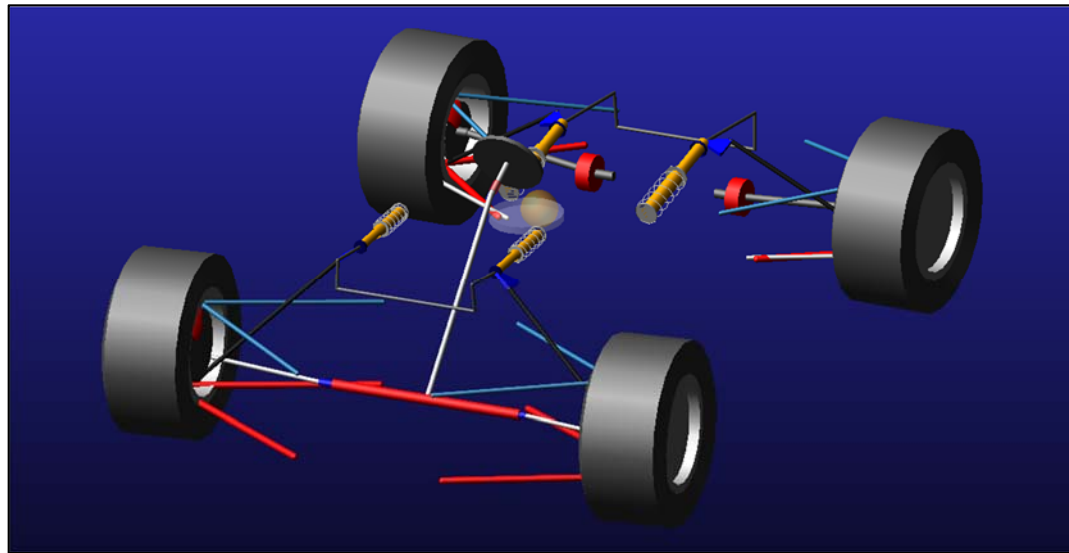
WORKSHOP 18

USING ADAMS INSIGHT WITH ADAMS CAR

Using Adams/Insight with Adams/Car

- **Problem Statement:**

- Use Adams/Insight together with Adams/Car to find out the effect of roll center on vehicle cornering capability by changing suspension geometry (hardpoints)



This workshop takes about one half hour to complete.

Using Adams/Insight with Adams/Car

- **In this workshop you will modify the suspension control arm hardpoints to optimize the roll axis and effect on cornering capability**

1. Front Suspension Analysis

- Simulate a parallel wheel travel on the Formula SAE front suspension assembly changing the inboard upper control arm points to get the desired front roll center heights using Adams/Insight.

2. Rear Suspension Analysis

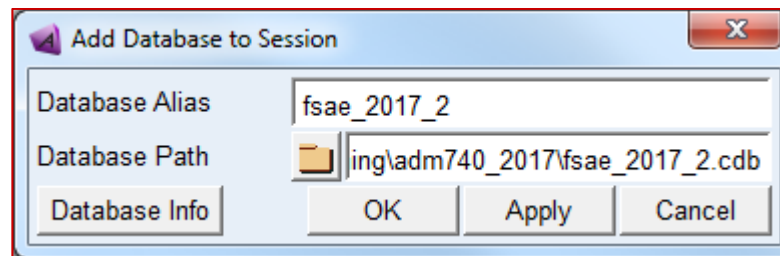
- Simulate a parallel wheel travel on the Formula SAE rear suspension assembly changing the inboard upper control arm points to get the desired rear roll center heights using Adams/Insight.

3. Full Vehicle Analysis

- Simulate Step Steer Full Vehicle Analysis to determine the effects of roll center height/inclination on vehicle turn-in response and overall cornering capacity using Adams Insight

Using Adams/Insight with Adams/Car

- **Add a new database to your session:**
 1. Start Adams/Car in Standard Interface.
 2. From the **Tools** menu, point to **Database Management**, and then select **Add to Session**.
 3. In the **Database Alias** text box, enter **fsae_2017_2**.
 4. In the **Database Path** field, click on the **Folder icon** and point to a **fsae_2017_2.cdb**.
 5. Click **OK**.

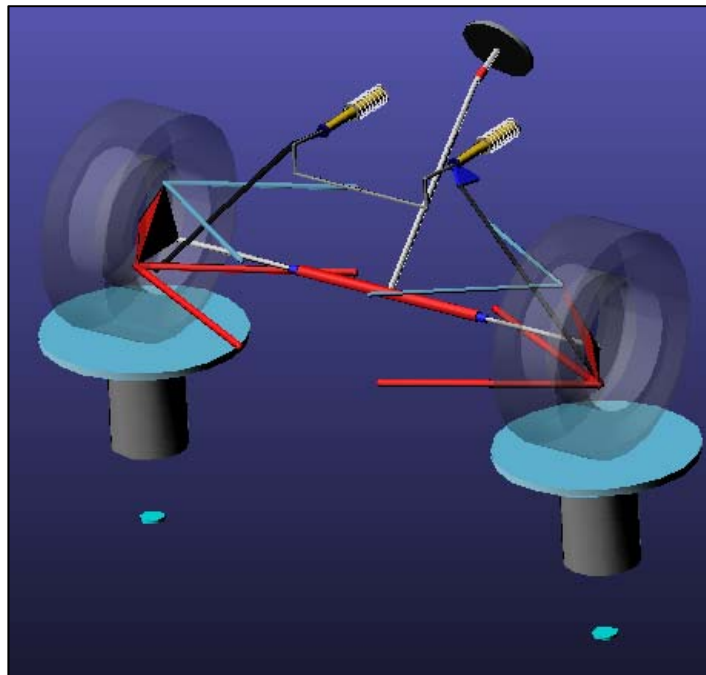


Front Suspension Analysis

- **Objective:**

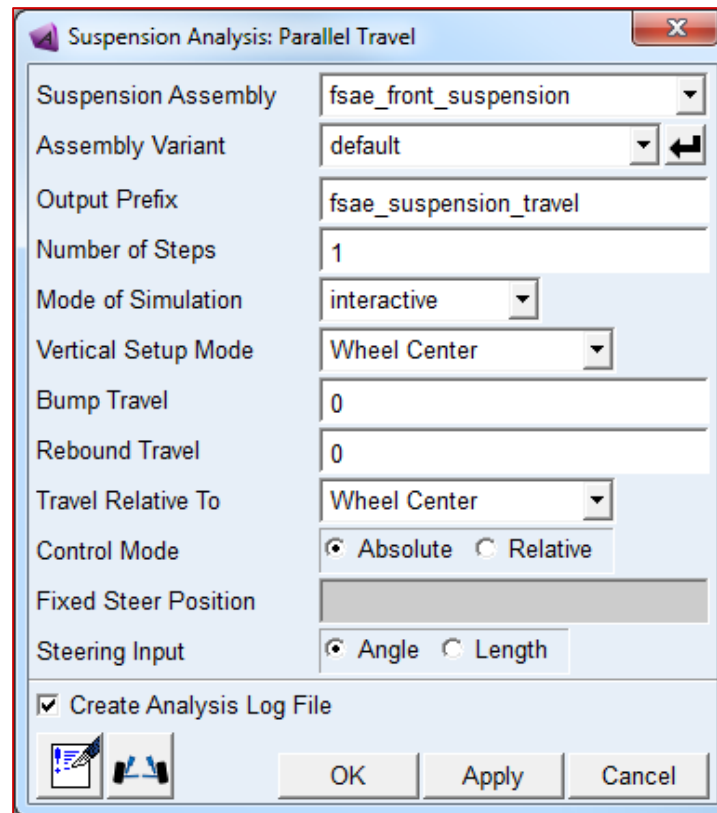
- Find inboard upper control arm points which will give the following roll center heights that we are going to test for cornering analysis

Roll center height = 0, 25, 50, 75, 100, 125, 150, 175, 200, 225



Front Suspension Analysis (cont.)

1. Open the **fsae_front_suspension** assembly from the **fsae_2017_2** database.
2. Simulate a parallel suspension travel as shown below:

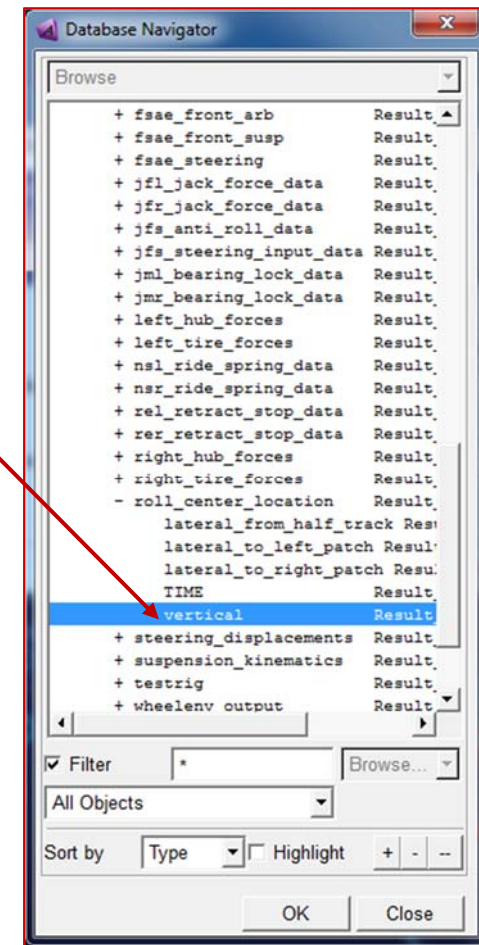
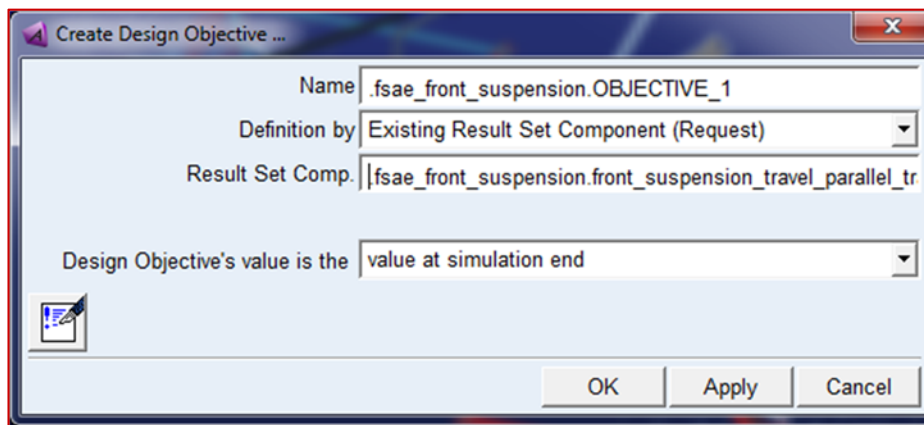
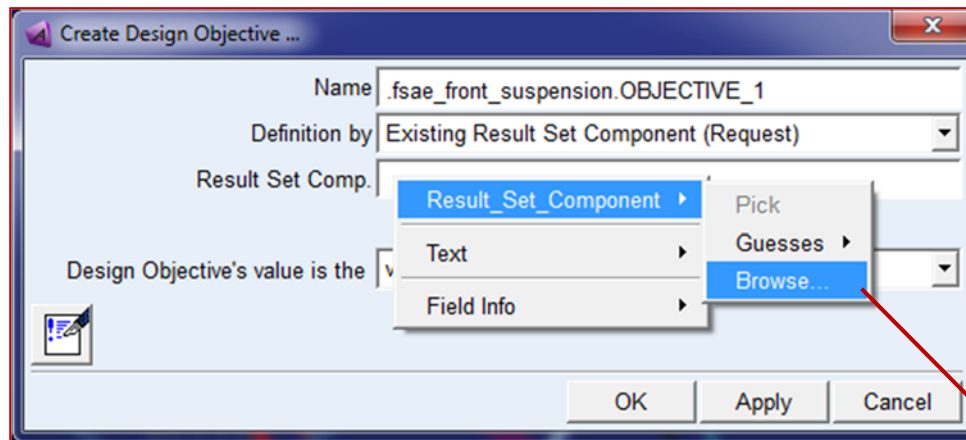


This assures that the suspension is at design position.

Front Suspension Analysis (cont.)

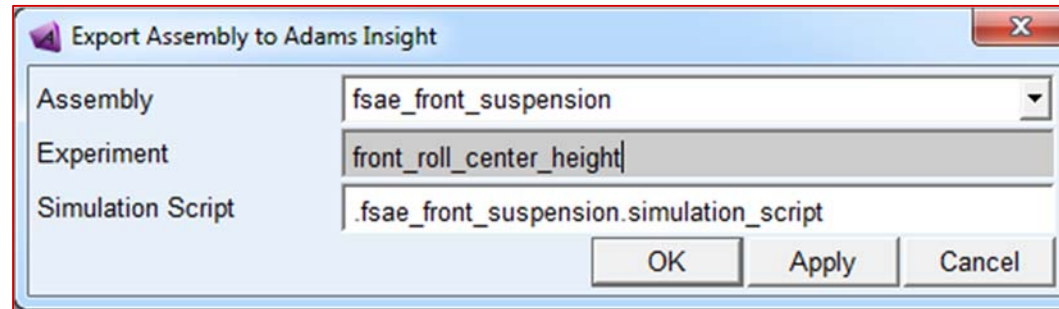
3. To Create Design Objective

- Go to **Simulate > DOE Interface > Design Objective > New...**
- Select result set component as roll center location vertical and Design objective value is at the end of simulation.



Front Suspension Analysis (cont.)


4. Export the Simulation to Adams/Insight
 - Go to **Simulate > DOE Interface > Adams/Insight > Export...**
 - Enter the details as per the dialog box shown below



Adams/Insight window opens with the new experiment called "front_roll_center_height".

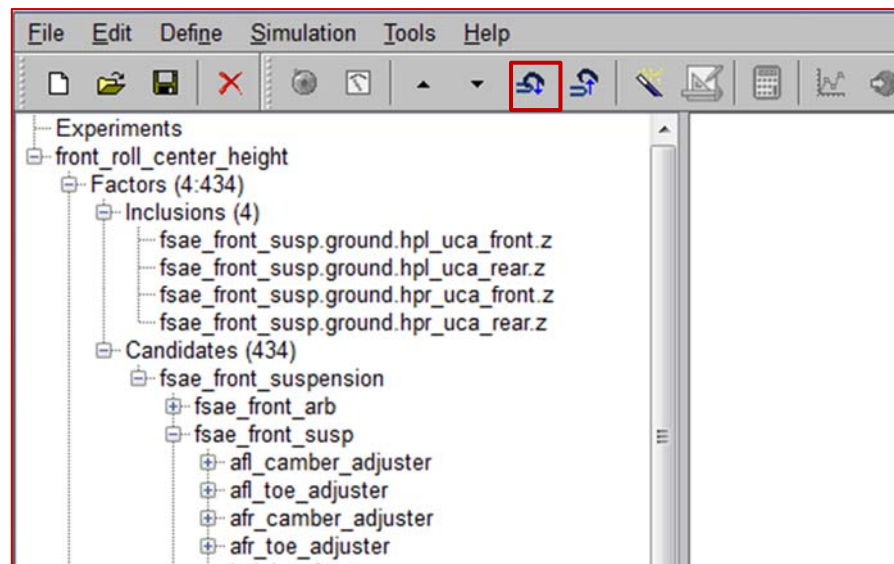
Front Suspension Analysis (cont.)

5. Promoting Factor:

- In the treeview, click the **+** in front of Factors.
- Select the **Candidate > fsae_front_suspension > fsae_front_susp > ground > hpl_uca_front > fsae_front_susp.ground.hpl_uca_front.z**
- Move your cursor to the Design Assistant toolbar and select the **Promote to inclusion tool** 
- Same way promote **hpr_uca_front.z**, **hpl_uca_rear.z**, **hpr_uca_rear.z** factors

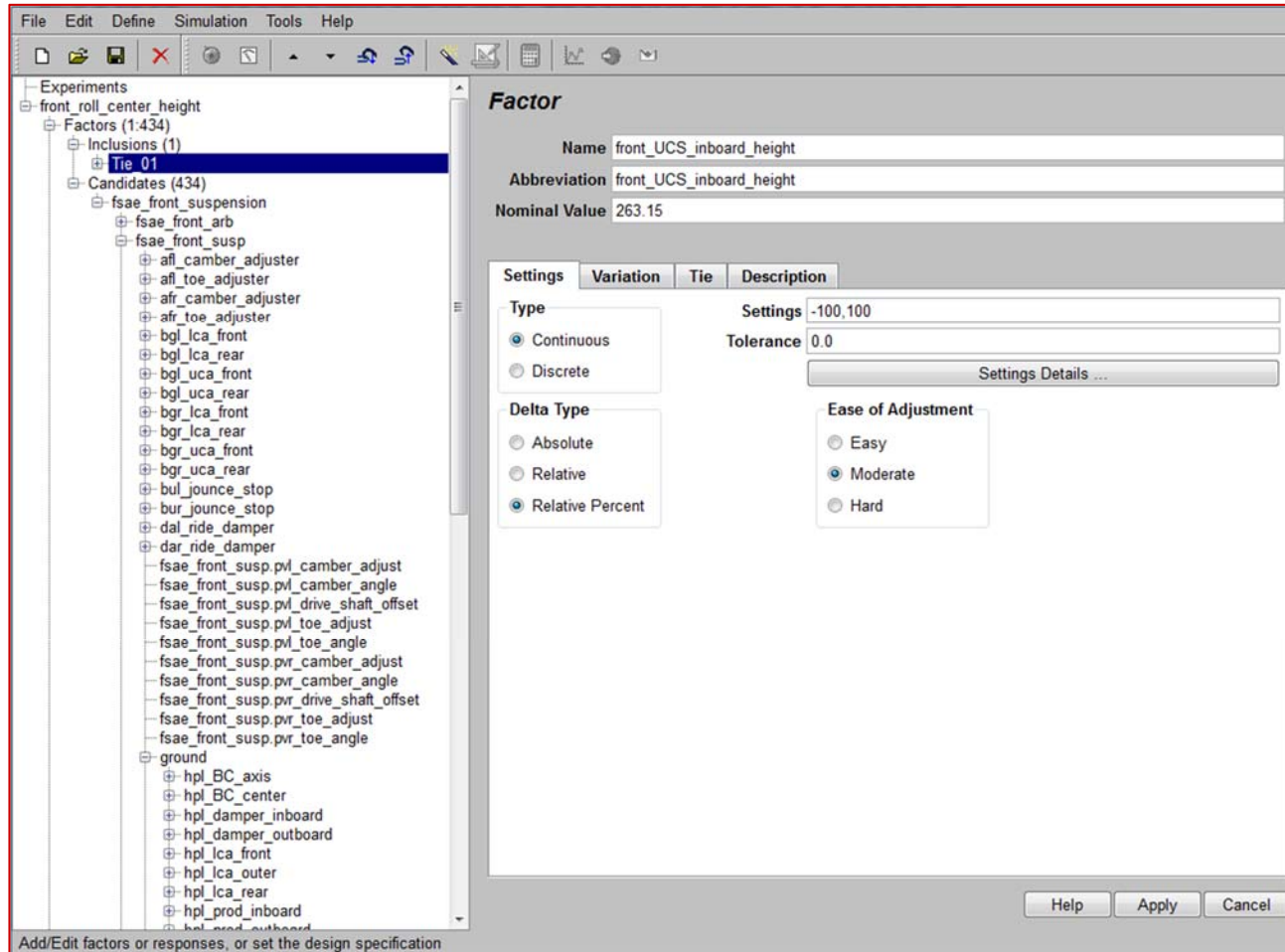
6. Tie the hardpoints:

- Highlight all of the points and press the tie button:



Front Suspension Analysis (cont.)

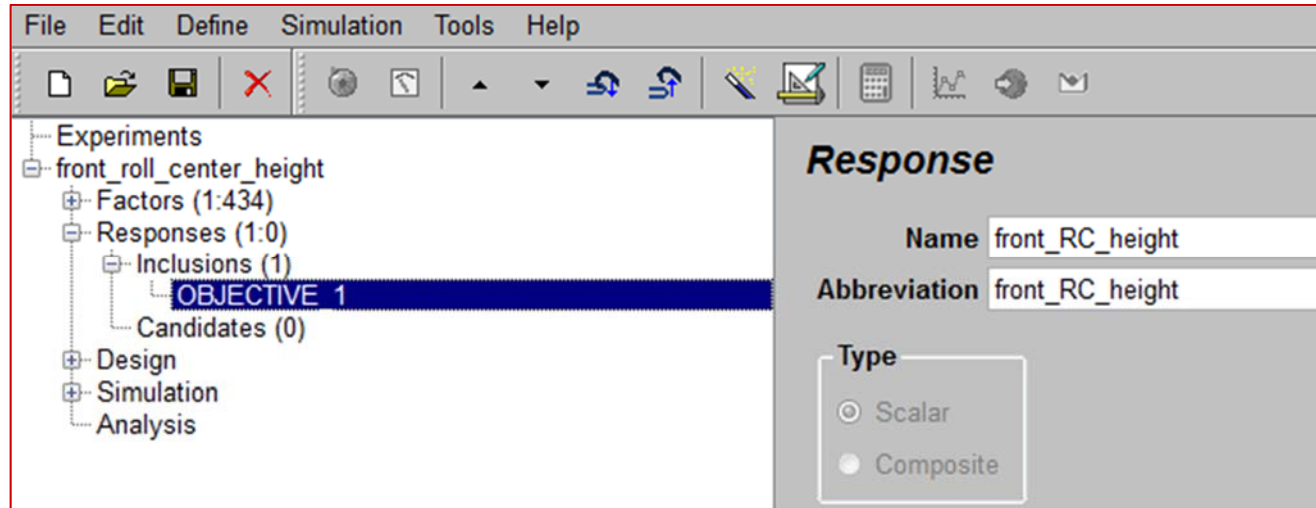
- Select **Tie_01** and Fill information as below image and click **Apply**



Front Suspension Analysis (cont.)


7. Promoting Response

- In the treeview, select the **+** in front of Responses.
- Under Candidates, you'll see **front_RC_height** and promote it.
- Make the abbreviation the same as the name and click **Apply**.



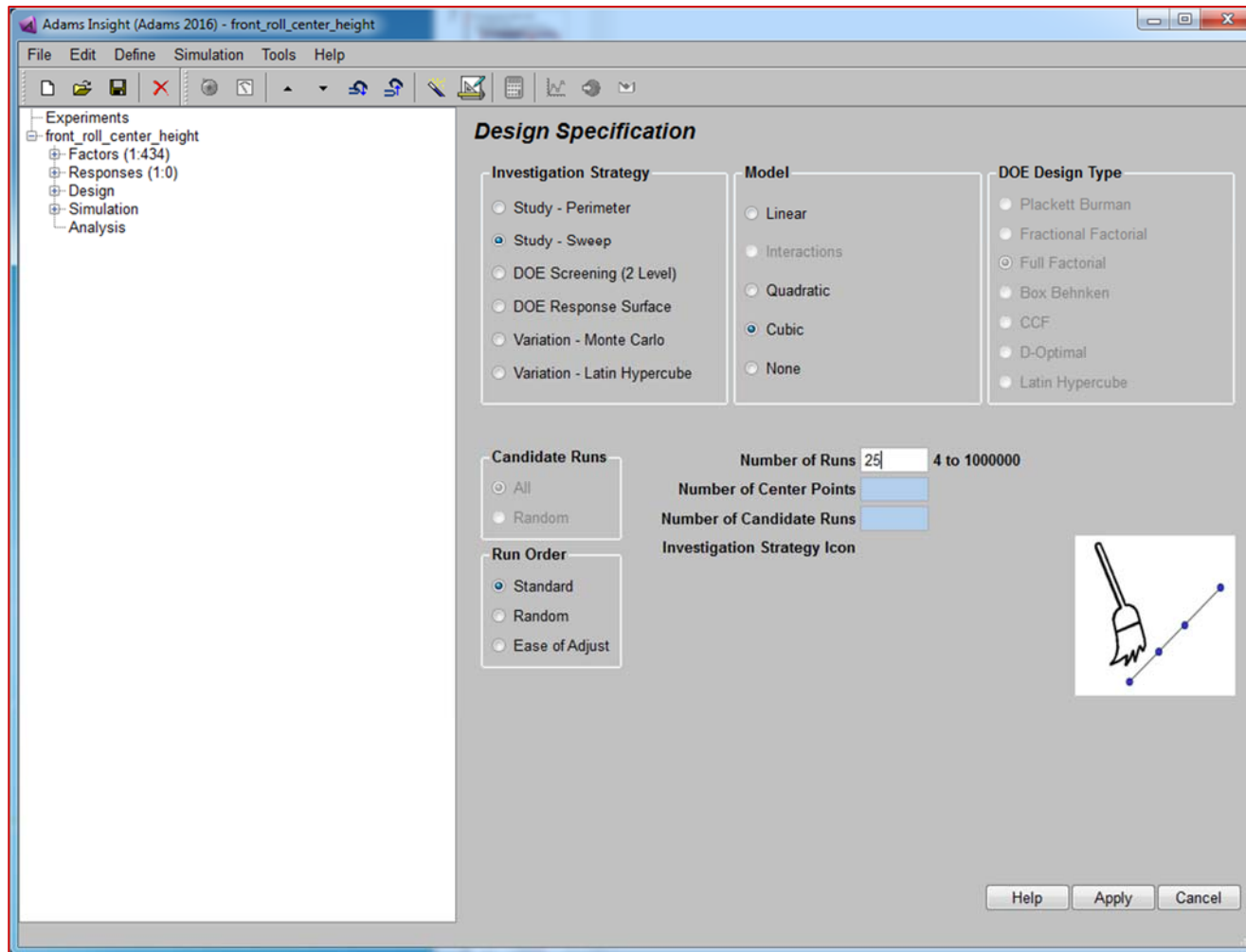
Front Suspension Analysis (cont.)

8. Setting Design Specifications

- In the Design assistant toolbar, click the **Set Design Specification Tool** , or in the treeview, expand the levels under Design, and then select Specification.
The Design Specification form appears in the viewport as shown on the next page.
- In the Design Specification form, make the following selections:
 - **Investigation Strategy:** **Study-Sweep**
 - **Model:** **Cubic**
 - **Number of runs:** **25**Use defaults for all remaining options.
- Select **Apply**.

Front Suspension Analysis (cont.)

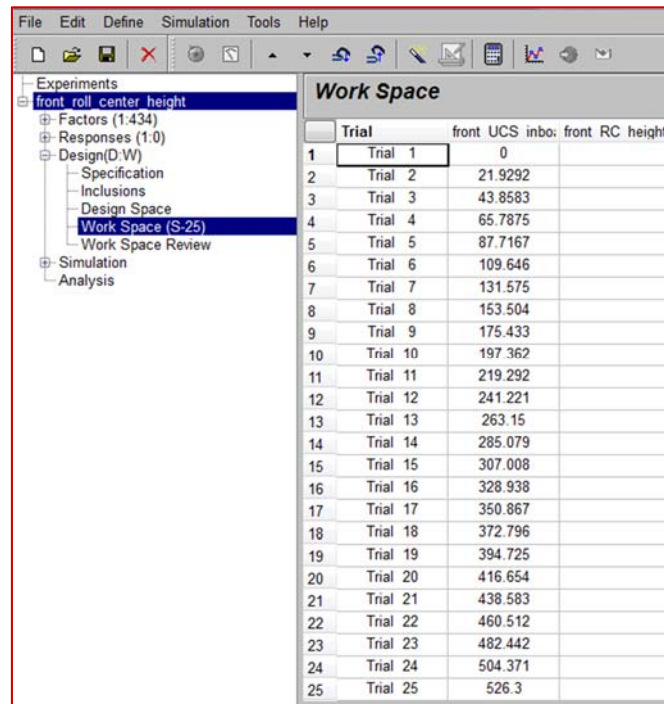
8. Setting Design Specifications (Cont.)



Front Suspension Analysis (cont.)

9. Generate Workspace

- In the Design Assistant toolbar, click the Generate Work Space tool  and verify the design matrix.



The screenshot shows the Design Assistant software interface. On the left is a treeview with the following structure:

- Experiments
 - front roll center height
 - Factors (1.434)
 - Responses (1.0)
 - Design(D.W)
 - Specification
 - Inclusions
 - Design Space
 - Work Space (S:25)
 - Work Space Review
 - Simulation
 - Analysis

On the right is a table titled 'Work Space' with the following data:


	Trial	front UCS inbo	front RC height
1	Trial 1	0	
2	Trial 2	21.9292	
3	Trial 3	43.8583	
4	Trial 4	65.7875	
5	Trial 5	87.7167	
6	Trial 6	109.646	
7	Trial 7	131.575	
8	Trial 8	153.504	
9	Trial 9	175.433	
10	Trial 10	197.362	
11	Trial 11	219.292	
12	Trial 12	241.221	
13	Trial 13	263.15	
14	Trial 14	285.079	
15	Trial 15	307.008	
16	Trial 16	328.938	
17	Trial 17	350.867	
18	Trial 18	372.796	
19	Trial 19	394.725	
20	Trial 20	416.654	
21	Trial 21	438.583	
22	Trial 22	460.512	
23	Trial 23	482.442	
24	Trial 24	504.371	
25	Trial 25	526.3	

Note: In the treeview, at the Design level, the letters D:W appear to indicate that the Design contains a successfully generated design work space.

Place your mouse pointer over column headings to display key information about the abbreviation shown.

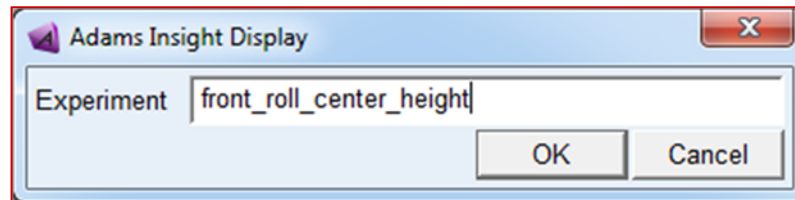
Front Suspension Analysis (cont.)

10. Running your experiment

- Once you've verified the information in the Work Space, you're ready to run the simulations
- Click **Build-Run-Load Adams View Simulations** 

11. Reviewing results

- After Adams completes the trials defined in your design matrix, you return to Adams/Insight interface to view the results.
- Go to Select **Simulate > DOE Interface > Adams/Insight > Display** and then fill with the same name of experiment, "front_roll_center_height".



Front Suspension Analysis (cont.)

- To view your simulation results:
 - In the treeview, under **Design**, select **Work Space**.
 - Simulation results from Adams/Car appear in the design matrix shown in below

Experiments

front_roll_center_height

Factors (1:434)

Responses (1:0)

Design(D:W)

Specification

Inclusions

Design Space

Work Space (S-25)

Work Space Review

Simulation

Analysis

Work Space

	Trial	front UCS inbo:	front RC height
1	Trial 1	0	245.763
2	Trial 2	21.9292	229.239
3	Trial 3	43.8583	212.222
4	Trial 4	65.7875	194.69
5	Trial 5	87.7167	176.617
6	Trial 6	109.646	157.98
7	Trial 7	131.575	138.751
8	Trial 8	153.504	118.901
9	Trial 9	175.433	98.4002
10	Trial 10	197.363	77.2155
11	Trial 11	219.292	55.3123
12	Trial 12	241.221	32.6534
13	Trial 13	263.15	9.19893
14	Trial 14	285.079	-15.0937
15	Trial 15	307.008	-40.2703
16	Trial 16	328.938	-66.38
17	Trial 17	350.867	-93.4757
18	Trial 18	372.796	-121.614
19	Trial 19	394.725	-150.857
20	Trial 20	416.654	-181.271
21	Trial 21	438.583	-212.927
22	Trial 22	460.512	-245.903
23	Trial 23	482.442	-280.283
24	Trial 24	504.371	-316.16
25	Trial 25	526.3	-353.633

Front Suspension Analysis (cont.)

12. Fitting Results

- Now that Adams/Car has completed the trials defined in your design matrix, you can use Adams/Insight to fit your results to a polynomial or a response surface.

The purpose of fitting your results is to establish a relationship between the factors and responses that you selected for the design matrix.


Fitting results includes a multiple regression. You will be able to investigate the parts of the regression in the Summary, located in the treeview under Analysis, after completing the following steps.

For more information on this topic, refer to the Adams/Insight online help.

Note: The material in the following sections includes statistical terms related to DOEs. For explanations of these terms, refer to the Adams/Insight online help.

Front Suspension Analysis (cont.)

- **To Fit Results:**





- From the Design Assistant toolbar, click the Fit Results tool . You can also select the Tools menu, and then click Fit New Model.

The Model Properties Summary window appears. Here, you can enter information on your model.

- In the Regression column, select the response, **front_RC_height**
- In the Display column, select the type of results you want to view. The following shows an example of the Fit table.




Front Suspension Analysis (Cont.)

- To Fit Results (Cont.):

Results Table with Fit for Regression					
Fit for regression "front_RC_height"					
	DOF	SS	MS	F	P
Model	3	7.96e+05	2.65e+05	3.52e+06	5.21e-60 
Error	21	1.58	0.0754		
Total	24	7.96e+05			
R2	1 				
R2adj	1 				
R/V	5.46e+03 				


For definitions of the items in the results tables, refer to the online help.

The tables also provide you with a color code that indicates the soundness of your results:

-  Green indicates that all fit criteria meet or exceed highest fitting thresholds
-  Yellow indicates that the fit criterion may bear investigation
-  Red indicates that the fit criterion should be investigated

Front Suspension Analysis (Cont.)

13. Optimize Results:

- From the Tools menu, select **Optimize Model** 

Optimize model 'Model_01'

Design Variables

	Minimum		Maximum	Value	Fixed
front_UCS_inboard_height	0		526.3	263.15	<input type="checkbox"/>

Design Objectives

	Minimum		Maximum	Value	Oper	Target	Weight	Cost
front_RC_height	-353.03		246.24	9.4797	Min	0	1	9.4797
Overall								9.4797

Optimize

Preferences... Reload Update Reset Write... Save Run

The window displays your model's factors and responses (design objectives). Only scalar responses are shown in the window. Composite responses are not displayed.

Front Suspension Analysis (Cont.)

13. Optimize Results(cont.):

- Here we are going to optimize model by changing design objectives (responses).
- In **Design Objective**
 1. Set **Oper** to **Equal**
 2. Set **Target** to **0**, which is our desired roll center
 3. Press **Run**
 4. Write down the value of **Design Variable** which is **271.61** as shown image on next page
- Now repeat above four steps for 25,50,75,100,125,150,175,200,225 target (roll center height) values and you will get results as per table shown on next page

Front Suspension Analysis (Cont.)

13. Optimize Results(cont.):

Optimize model 'Model_01'

Design Variables

	Minimum	Maximum	Value	Fixed
front_UCS_inboard_height	0	526.3	271.83	<input type="checkbox"/>

Design Objectives

	Minimum	Maximum	Value	Op	Targ	Weight	Cost
front_RC_height	-353.03	246.24	-0.029059	Equal	0	1	
Overall							0

Optimize

Roll Center Height:	UCA_Inboard_Height:
0	271.83
25	248.74
50	224.73
75	199.75
100	173.76
125	146.71
150	118.57
175	89.314
200	58.95
225	27.509

Front Suspension Analysis (Cont.)

14. Validate Results:

- It is always good to go back and validate that the points the optimize tool puts out will yield the results you expect.
- For example check the upper control arm inboard height of 199.75mm and it yielded a roll center height of 74.975mm instead of 75mm. This is because we fit a curve that was close to linear, however as you fit more nonlinear systems, you will have more variation.

Rear Suspension Analysis

- **Objective:**

- Find inboard upper control arm points which will give the following roll center heights that we are going to test for cornering analysis

Roll Center Height = 0, 50, 100, 150, 200, 250, 300, 350, 400

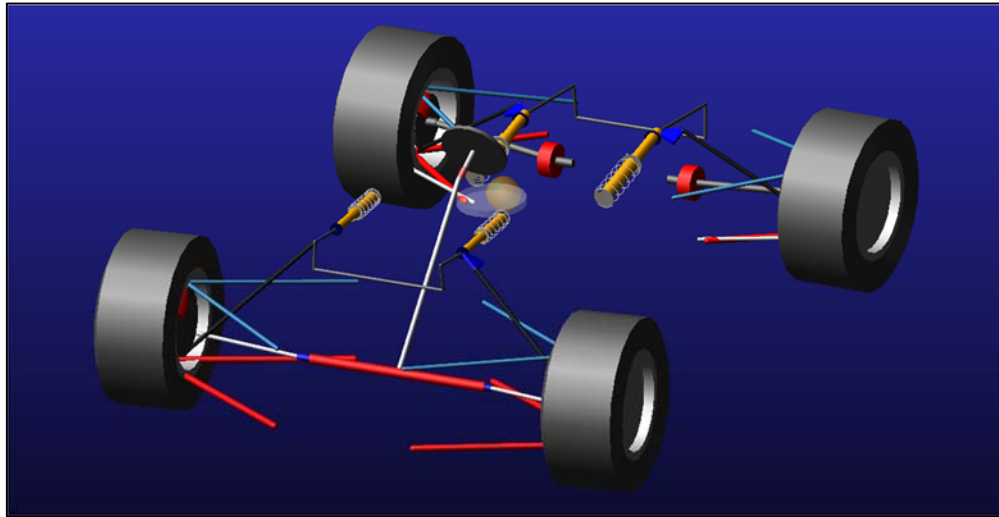
- **Procedure:**

- Open the **fsae_rear_suspension** assembly from the **fsae_2017_2** database.
- Perform all the steps for the rear suspension assembly similar to the front suspension with experiment name as “rear_roll_center_height”
- Perform optimization for above mentioned roll center height to get the results shown here

Roll Center Height:	UCA_Inboard_Height:
0	287.37
50	258.14
100	228.26
150	197.71
200	166.46
250	134.49
300	101.77
350	68.289
400	34.013

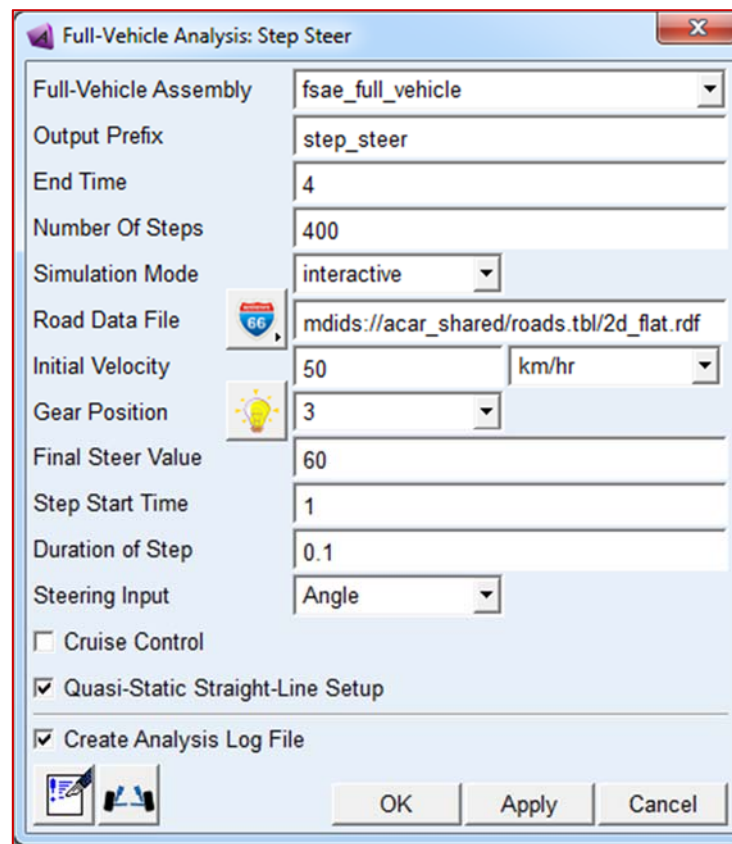
Full Vehicle Analysis

- **Objective:**
 - Find out effect of roll center height on vehicle cornering capability for calculated front and rear UCA hardpoints



Full Vehicle Analysis

1. Open the **fsae_full_vehicle** assembly from the **fsae_2017_2** database
2. Simulate a Step Steer Analysis as shown below:
 - Select **Simulate** → **Full-Vehicle Analysis** → **Open-Loop Steering Events** → **Step Steer...**
 - Fill out the dialog box as shown:



Full-Vehicle Analysis: Step Steer

Full-Vehicle Assembly: fsae_full_vehicle

Output Prefix: step_steer

End Time: 4

Number Of Steps: 400

Simulation Mode: interactive

Road Data File: mdids://acar_shared/roads.tbl/2d_flat.rdf

Initial Velocity: 50 km/hr

Gear Position: 3

Final Steer Value: 60

Step Start Time: 1

Duration of Step: 0.1

Steering Input: Angle

☐ Cruise Control

☒ Quasi-Static Straight-Line Setup

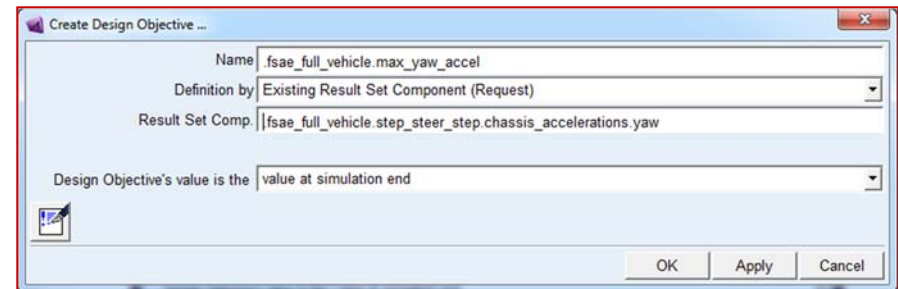
☒ Create Analysis Log File

OK Apply Cancel

Full Vehicle Analysis

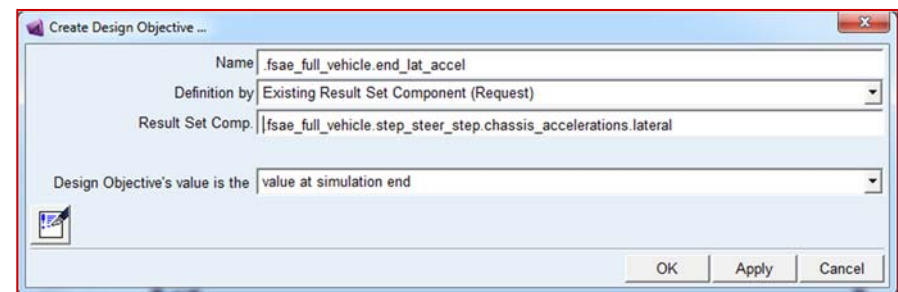
3. Create Design Objective

- Design Objective for Yaw Acceleration:
- Select result set component as chassis yaw acceleration and Design objective value is maximum absolute value during simulation



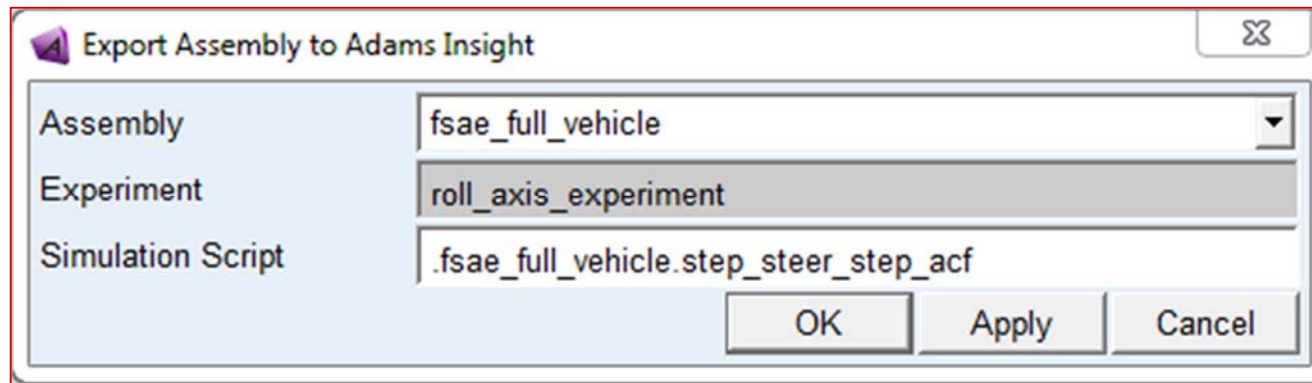
4. Create Design Objective (cont.)

- Design Objective for Lateral Acceleration:
- Select result set component as chassis lateral acceleration and Design objective value is value at simulation end



Full Vehicle Analysis (Cont.)

5. Export the simulation to Adams/Insight
 - Go to **Simulate > DOE Interface > Adams/Insight > Export...**
 - Enter the details as per the dialog box shown below



Adams/Insight window opens with the new experiment called "roll_axis_experiment".

Full Vehicle Analysis (Cont.)

6. Promoting Factor:

- In the treeview, click the **+** in front of Factors.
- Select the **Candidate > fsae_full_vehicle > fsae_front_susp > ground > hpl_uca_front > fsae_front_susp.ground.hpl_uca_front.z**
- Move your cursor to the **Design Assistant toolbar** and select the **Promote to inclusion tool**
- Same way promote **hpr_uca_front.z, hpl_uca_rear.z, hpr_uca_rear.z** factors for **front and rear** suspension

7. Tie the hardpoints:

- Create two ties (**Front_UCA_Inboard_Height** and **Rear_UCA_Inboard_Height**) with following front and rear inclusion factors

Full Vehicle Analysis (Cont.)

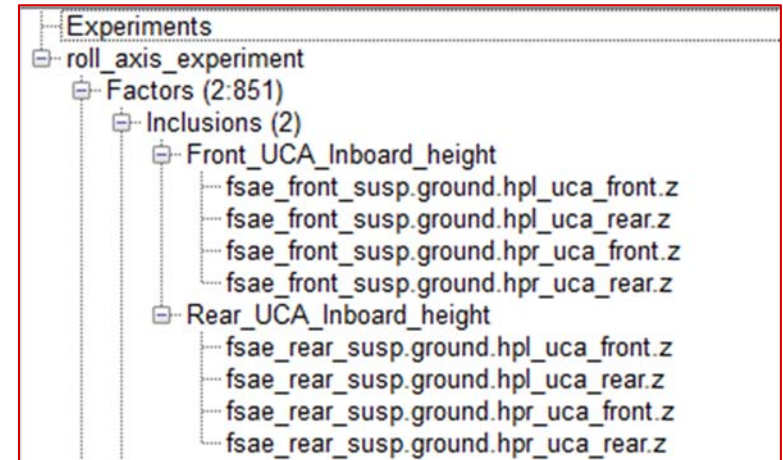
Add the following hardpoints under the following ties:

– Front_UCA_Inboard_Height

- fsae_front_susp.ground.hpl_uca_front.z
- fsae_front_susp.ground.hpl_uca_rear.z
- fsae_front_susp.ground.hpr_uca_front.z
- fsae_front_susp.ground.hpr_uca_rear.z

– Rear_UCA_Inboard_Height

- fsae_rear_susp.ground.hpl_uca_front.z
- fsae_rear_susp.ground.hpl_uca_rear.z
- fsae_rear_susp.ground.hpr_uca_front.z
- fsae_rear_susp.ground.hpr_uca_rear.z



Full Vehicle Analysis (Cont.)

- Now click **Front_UCA_Inboard_Height** and fill the dialog box as shown below with the points we found for the front suspension to give the desired front roll center heights:

The screenshot shows the MSC Software Factor dialog box. On the left is a tree view of the project structure. The main panel on the right is titled 'Factor' and contains the following fields and sections:

- Name:** Front_UCA_Inboard_height
- Abbreviation:** Front_UCA_Inboard_height
- Nominal Value:** 1
- Settings** tab is selected, showing:
 - Type:** Discrete (selected)
 - Delta Type:** Absolute (selected)
 - Ease of Adjustment:** Moderate (selected)
 - Settings:** 271.61, 248.5, 224.49, 199.5, 173.5, 146.44, 118.28, 89.018, 58.643, 27.19
 - Tolerance:** 0.0
 - Settings Details ...** button

The tree view on the left shows the following structure:

- Experiments
 - roll_axis_experiment
 - Factors (2:851)
 - Inclusions (2)
 - Front_UCA_Inboard_height
 - fsae_front_susp.ground.hpl_uca_front.z
 - fsae_front_susp.ground.hpl_uca_rear.z
 - fsae_front_susp.ground.hpr_uca_front.z
 - fsae_front_susp.ground.hpr_uca_rear.z
 - Rear_UCA_Inboard_height
 - fsae_rear_susp.ground.hpl_uca_front.z
 - fsae_rear_susp.ground.hpl_uca_rear.z
 - fsae_rear_susp.ground.hpr_uca_front.z
 - fsae_rear_susp.ground.hpr_uca_rear.z
 - Candidates (851)
 - Responses (4:34)
 - Design
 - Simulation
 - Analysis

Full Vehicle Analysis (Cont.)

- Click **Rear_UCA_Inboard_Height** and fill the dialogue box as shown below with the points we found for the rear suspension to give the desired rear roll center heights:

The screenshot shows the MSC Software interface. On the left is a tree view of the project structure. The 'Factors' folder is expanded, showing 'Inclusions (2)'. Under 'Inclusions', 'Front_UCA_Inboard_height' and 'Rear_UCA_Inboard_height' are listed. 'Rear_UCA_Inboard_height' is selected and highlighted in blue. Below it, several specific points are listed, including 'fsae_rear_susp.ground.hpl_uca_rear.z'. On the right is the 'Factor' dialog box. It has fields for 'Name' (Rear_UCA_Inboard_height), 'Abbreviation' (Rear_UCA_Inboard_height), and 'Nominal Value' (1). Below these are tabs for 'Settings', 'Variation', 'Tie', and 'Description'. The 'Settings' tab is active, showing 'Type' (Discrete), 'Delta Type' (Absolute), and 'Ease of Adjustment' (Moderate). The 'Settings' field contains a list of values: 287.37, 258.14, 228.26, 197.71, 166.46, 134.49, 101.77, 68.289, 34.013. The 'Tolerance' field is set to 0.0. A 'Settings Details ...' button is at the bottom right of the dialog.

Factor

Name: Rear_UCA_Inboard_height

Abbreviation: Rear_UCA_Inboard_height

Nominal Value: 1

Settings | Variation | Tie | Description

Type: ☐ Continuous ☒ Discrete

Delta Type: ☒ Absolute ☐ Relative ☐ Relative Percent

Ease of Adjustment: ☐ Easy ☒ Moderate ☐ Hard

Settings: 287.37, 258.14, 228.26, 197.71, 166.46, 134.49, 101.77, 68.289, 34.013

Tolerance: 0.0

Settings Details ...

Full Vehicle Analysis (Cont.)

8. Promoting Response

- In the treeview, select the **+** in front of Responses.
- Under **Candidates**, you'll see **end_lat_accel** and promote it.
- Make the abbreviation the same as the name and click **Apply**.

The screenshot displays the MSC Software interface. On the left, a treeview under 'Experiments' shows 'roll_axis_experiment' expanded. Within it, 'Responses (2:36)' is expanded, and 'end lat accel' is selected. The right panel, titled 'Response', shows the 'Name' and 'Abbreviation' fields both set to 'end_lat_accel'. The 'Type' section has 'Scalar' selected with a radio button.

Response	
Name	end_lat_accel
Abbreviation	end_lat_accel
Type	<input checked="" type="radio"/> Scalar <input type="radio"/> Composite

Full Vehicle Analysis (Cont.)

8. Promoting Response (cont.)


- In the treeview, select the **+** in front of Responses.
- Under **Candidates**, you'll see **max_yaw_ccel** and promote it.
- Make the abbreviation the same as the name and click **Apply**.

The screenshot displays the MSC Software interface. On the left, a treeview shows the hierarchy: Experiments > roll_axis_experiment > Factors (2:851) > Candidates (851). The 'max_yaw accel' candidate is selected and highlighted in blue. On the right, the 'Response' configuration panel is visible. It contains the following fields:

- Name:** max_yaw_accel
- Abbreviation:** max_yaw_accel
- Type:** Radio buttons for 'Scalar' (selected) and 'Composite'.

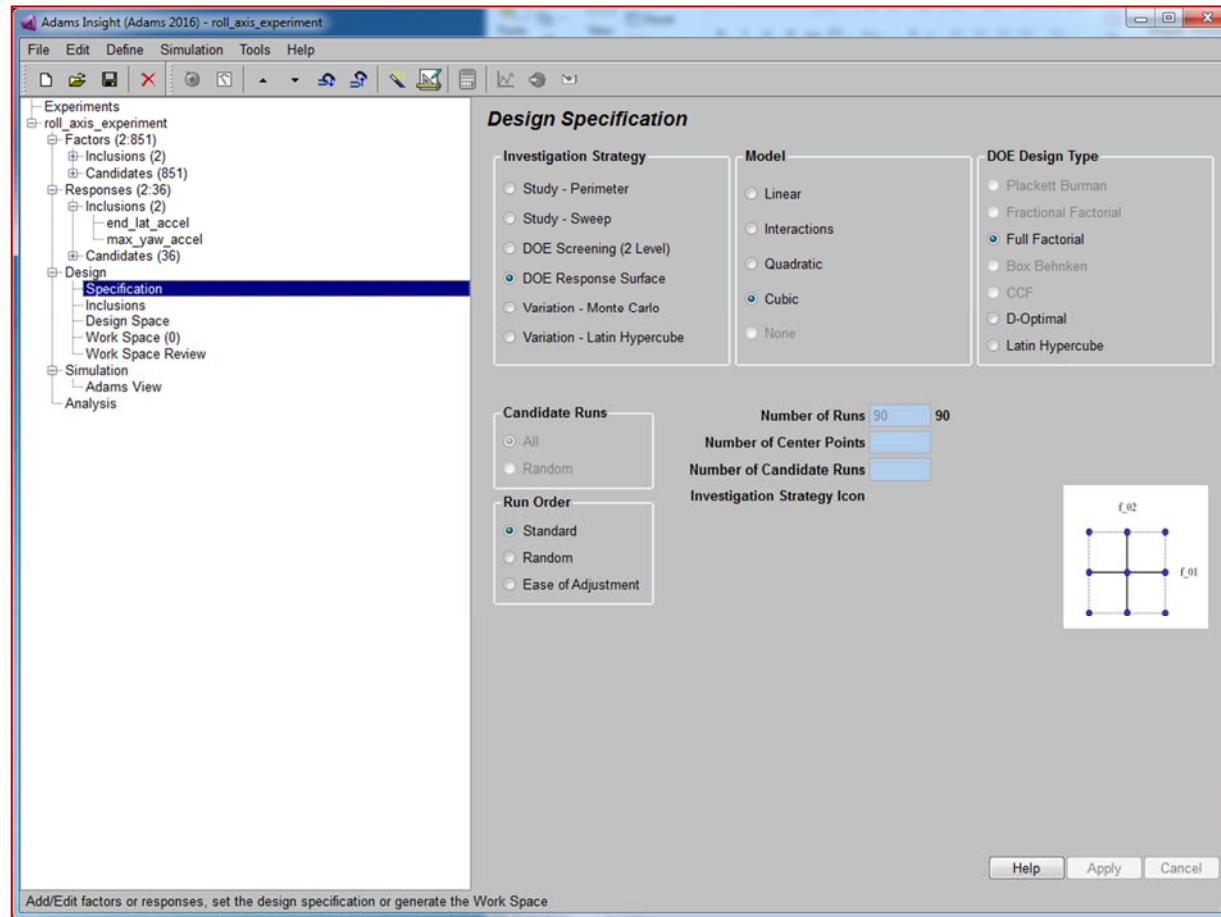
Full Vehicle Analysis (Cont.)

9. Setting Design Specifications

- In the **Design assistant toolbar**, click the **Set Design Specification Tool** , or in the treeview, expand the levels under **Design**, and then select **Specification**.
The Design Specification form appears in the viewport as shown on the next page.
- In the Design Specification form, make the following selections:
 - **Investigation Strategy:** **DOE Response Surface**
 - **Model:** **Cubic**
 - **DOE Design Type:** **Full Factorial**Use defaults for all remaining options.
- Select **Apply**


Full Vehicle Analysis (Cont.)

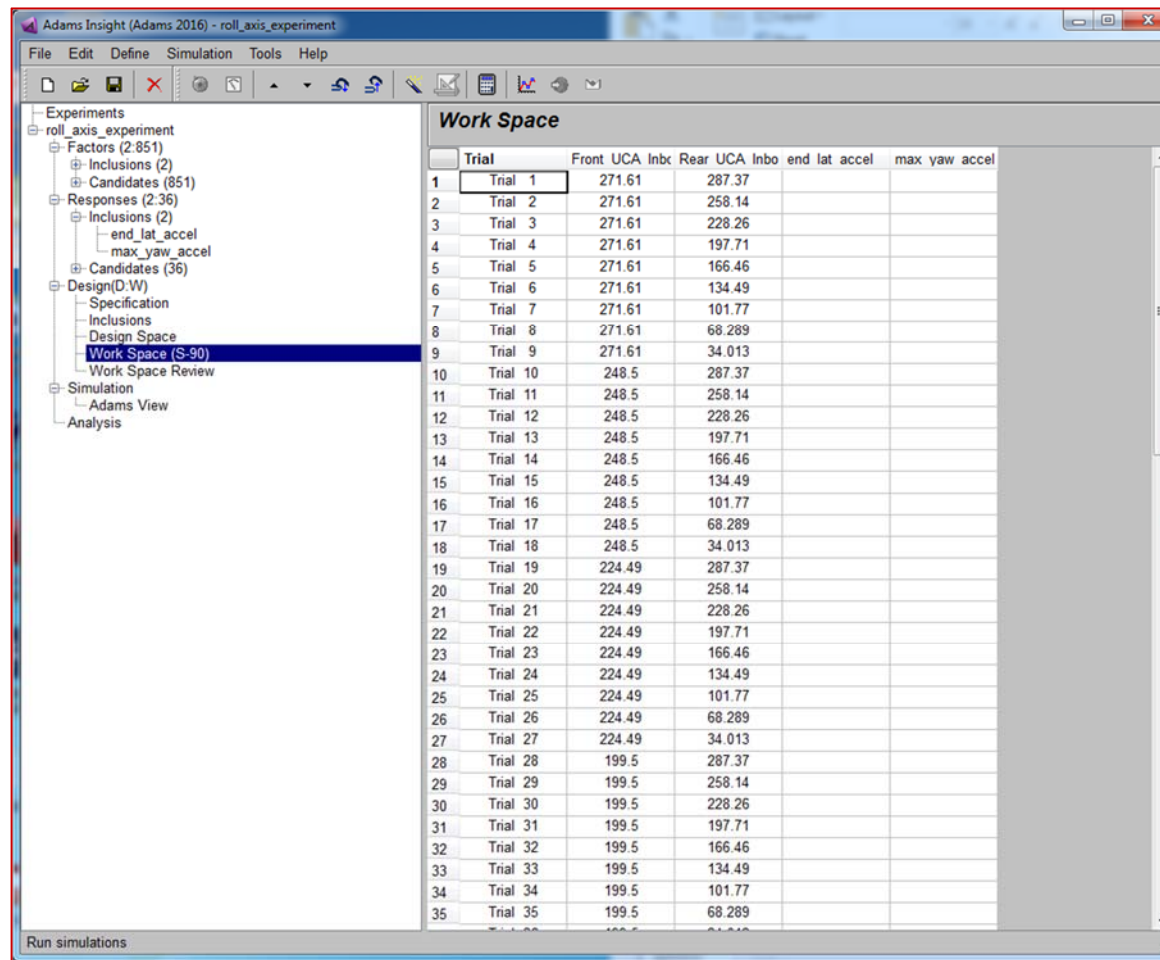
9. Setting Design Specifications (cont.)



Full Vehicle Analysis (Cont.)

10. Generate Workspace

- In the Design Assistant toolbar, click the Generate Work Space tool , and verify the design matrix.




The screenshot shows the Adams Insight (Adams 2016) - roll_axis_experiment window. The left pane displays a tree view of the experiment structure, with 'Work Space (S-90)' selected. The right pane displays a table titled 'Work Space' with 35 trials and 6 columns: Trial, Front UCA, Inbx, Rear UCA, Inbo, and end lat accel. The table data is as follows:

Trial	Front UCA	Inbx	Rear UCA	Inbo	end lat accel
1	271.61		287.37		
2	271.61		258.14		
3	271.61		228.26		
4	271.61		197.71		
5	271.61		166.46		
6	271.61		134.49		
7	271.61		101.77		
8	271.61		68.289		
9	271.61		34.013		
10	248.5		287.37		
11	248.5		258.14		
12	248.5		228.26		
13	248.5		197.71		
14	248.5		166.46		
15	248.5		134.49		
16	248.5		101.77		
17	248.5		68.289		
18	248.5		34.013		
19	224.49		287.37		
20	224.49		258.14		
21	224.49		228.26		
22	224.49		197.71		
23	224.49		166.46		
24	224.49		134.49		
25	224.49		101.77		
26	224.49		68.289		
27	224.49		34.013		
28	199.5		287.37		
29	199.5		258.14		
30	199.5		228.26		
31	199.5		197.71		
32	199.5		166.46		
33	199.5		134.49		
34	199.5		101.77		
35	199.5		68.289		

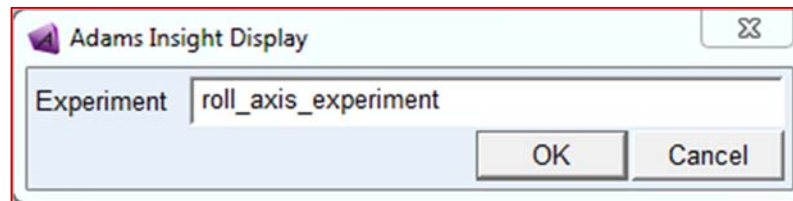
Full Vehicle Analysis (Cont.)

11. Running your experiment

- Once you've verified the information in the Work Space, you're ready to run the simulations
- Click **Build-Run-Load Adams View Simulations** 

12. Reviewing results

- After Adams completes the trials defined in your design matrix, you return to Adams/Insight interface to view the results.
- Go to Select **Simulate > DOE Interface > Adams/Insight > Display** and then fill with the same name of experiment, "**roll_axis_experiment**".



Full Vehicle Analysis (Cont.)

- To view your simulation results:
 - In the treeview, under Design, select Work Space.

	Trial	Front UCA Inbc	Rear UCA Inbo en
1	Trial 1	271.61	287.37
2	Trial 2	271.61	258.14
3	Trial 3	271.61	228.26
4	Trial 4	271.61	197.71
5	Trial 5	271.61	166.46
6	Trial 6	271.61	134.49
7	Trial 7	271.61	101.77
8	Trial 8	271.61	68.289
9	Trial 9	271.61	34.013
10	Trial 10	248.5	287.37
11	Trial 11	248.5	258.14
12	Trial 12	248.5	228.26
13	Trial 13	248.5	197.71
14	Trial 14	248.5	166.46
15	Trial 15	248.5	134.49
16	Trial 16	248.5	101.77
17	Trial 17	248.5	68.289
18	Trial 18	248.5	34.013
19	Trial 19	224.49	287.37
20	Trial 20	224.49	258.14
21	Trial 21	224.49	228.26
22	Trial 22	224.49	197.71
23	Trial 23	224.49	166.46
24	Trial 24	224.49	134.49
25	Trial 25	224.49	101.77
26	Trial 26	224.49	68.289
27	Trial 27	224.49	34.013
28	Trial 28	199.5	287.37
29	Trial 29	199.5	258.14
30	Trial 30	199.5	228.26
31	Trial 31	199.5	197.71
32	Trial 32	199.5	166.46
33	Trial 33	199.5	134.49
34	Trial 34	199.5	101.77
35	Trial 35	199.5	68.289
36	Trial 36	199.5	34.013
37	Trial 37	173.5	287.37
38	Trial 38	173.5	258.14
39	Trial 39	173.5	228.26
40	Trial 40	173.5	197.71
41	Trial 41	173.5	166.46
42	Trial 42	173.5	134.49
43	Trial 43	173.5	101.77
44	Trial 44	173.5	68.289
45	Trial 45	173.5	34.013
46	Trial 46	146.44	287.37
47	Trial 47	146.44	258.14
48	Trial 48	146.44	228.26
49	Trial 49	146.44	197.71

49	Trial 49	146.44	197.71
50	Trial 50	146.44	166.46
51	Trial 51	146.44	134.49
52	Trial 52	146.44	101.77
53	Trial 53	146.44	68.289
54	Trial 54	146.44	34.013
55	Trial 55	118.28	287.37
56	Trial 56	118.28	258.14
57	Trial 57	118.28	228.26
58	Trial 58	118.28	197.71
59	Trial 59	118.28	166.46
60	Trial 60	118.28	134.49
61	Trial 61	118.28	101.77
62	Trial 62	118.28	68.289
63	Trial 63	118.28	34.013
64	Trial 64	89.018	287.37
65	Trial 65	89.018	258.14
66	Trial 66	89.018	228.26
67	Trial 67	89.018	197.71
68	Trial 68	89.018	166.46
69	Trial 69	89.018	134.49
70	Trial 70	89.018	101.77
71	Trial 71	89.018	68.289
72	Trial 72	89.018	34.013
73	Trial 73	58.643	287.37
74	Trial 74	58.643	258.14
75	Trial 75	58.643	228.26
76	Trial 76	58.643	197.71
77	Trial 77	58.643	166.46
78	Trial 78	58.643	134.49
79	Trial 79	58.643	101.77
80	Trial 80	58.643	68.289
81	Trial 81	58.643	34.013
82	Trial 82	27.19	287.37
83	Trial 83	27.19	258.14
84	Trial 84	27.19	228.26
85	Trial 85	27.19	197.71
86	Trial 86	27.19	166.46
87	Trial 87	27.19	134.49
88	Trial 88	27.19	101.77
89	Trial 89	27.19	68.289
90	Trial 90	27.19	34.013

Full Vehicle Analysis (Cont.)

13. The Fit Tables will look like shown in below figure:

Results Table with Fit for Regression

Fit for regression "end_lat_accel"						Fit for regression "max_yaw_accel"					
	DOF	SS	MS	F	P		DOF	SS	MS	F	P
Model	9	0.265	0.0294	47.4	2.06e-28	Model	9	5.06e+04	5.63e+03	34.4	5.84e-24
Error	80	0.0496	0.00062			Error	80	1.31e+04	163		
Total	89	0.314				Total	89	6.37e+04			
R2	0.842	?				R2	0.795	✖			
R2adj	0.824	?				R2adj	0.772	?			
R/V	38.9	●				R/V	24.1	●			

For definitions of the items in the results tables, refer to the online help.

The tables also provide you with a color code that indicates the soundness of your results:

- Green indicates that all fit criteria meet or exceed highest fitting thresholds
- ? Yellow indicates that the fit criterion may bear investigation
- ✖ Red indicates that the fit criterion should be investigated

Full Vehicle Analysis (Cont.)

- To fit your results for end_lat_accel and max_yaw_accel:
 - Go to **Tools > Refine Model Manually > Remove Outliners...**
 - Select the **end_lat_accel** from **Regressions** section
 - Enter the following runs in **Removed Runs** section:
7,8,9,17,18,27,36,37,39,45,46,48,54,55,57,63,72
 - Click **Apply**
 - Select the **max_yaw_accel** from **Regressions** section
 - Enter the following runs in **Removed Runs** section:
3,5,6,13,19,20,21,29,38,39,41,46,47,48,55,56,57,58,64,65,66,67,73,74,75,76,83,85,87
 - Click **OK**

Full Vehicle Analysis (Cont.)


- After removing outliers Fit Tables will look like below figure

Fit for regression "end_lat_accel"					
	DOF	SS	MS	F	P
Model	9	0.0302	0.00335	314	8.07e-49 ●
Error	63	0.000672	1.07e-05		
Total	72	0.0308			
R2	0.978 ●				
R2adj	0.975 ●				
R/V	57.4 ●				

Fit for regression "max_yaw_accel"					
	DOF	SS	MS	F	P
Model	9	2.87e+04	3.18e+03	240	1.64e-38 ●
Error	51	677	13.3		
Total	60	2.93e+04			
R2	0.977 ●				
R2adj	0.973 ●				
R/V	61.4 ●				

Full Vehicle Analysis (Cont.)

14. Publishing Results

- Adams/Insight lets you save your files as either HTML or SYLK files. Once saved, you can use either a browser or spreadsheet program, such as Excel, to modify factors and see the effect on responses without performing full simulations.
- To Publish Results:
 - In the treeview, under **Analysis**, select **Model_01**, go to the **Design Assistant toolbar**, and then select the **Export to Web, slk, etc. tool** . You can also select the **Tools** menu, point to **Export**, and then click **HTML Format**.
 - The Save dialog box appears and prompts you to save your results as xxx.htm, where xxx is the name of your file.
 - Enter a name for your file and specify the path where you would like it to reside, and then click **Save**.
 - Adams/Insight saves your file in the directory that you specified.

Full Vehicle Analysis (Cont.)

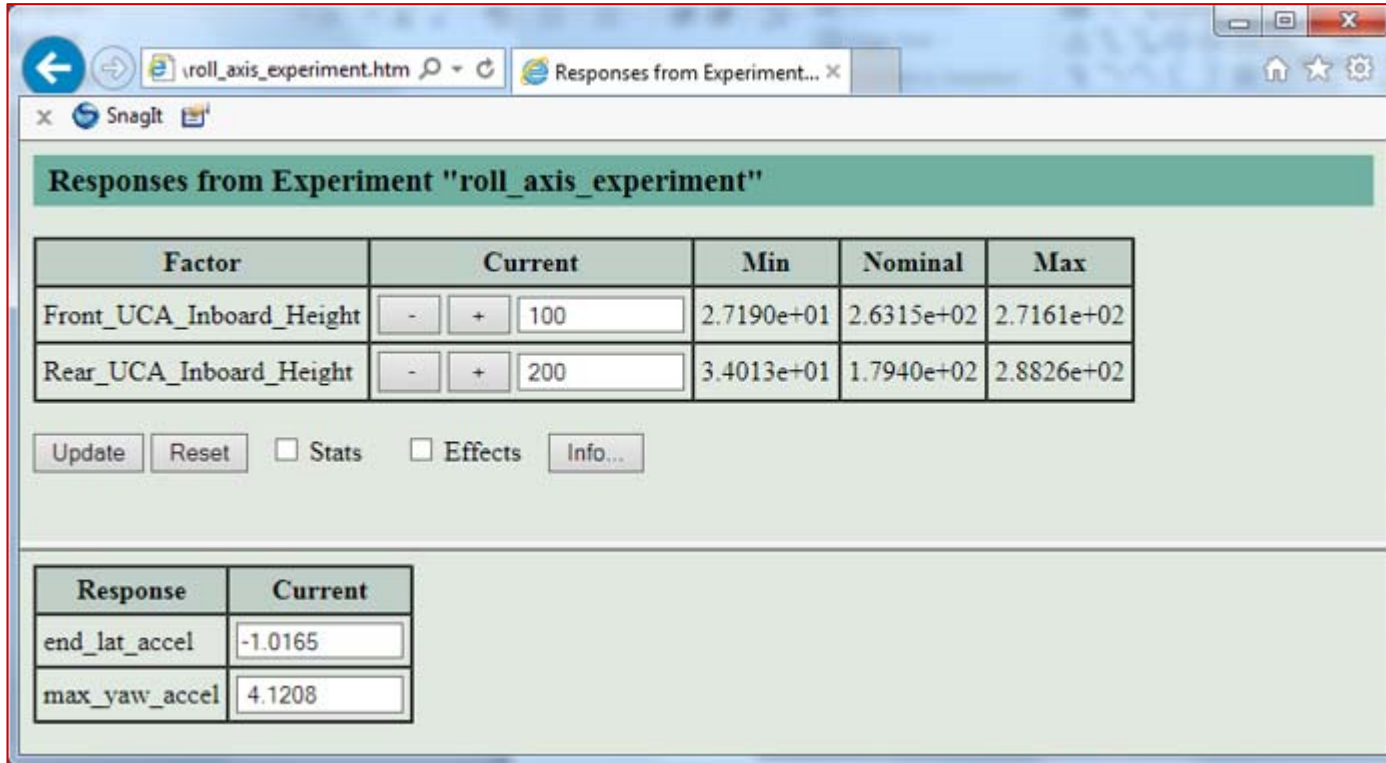
– Modifying Values using a Web browser

Using the HTML page that you saved (see steps above), you can modify the input factor values of your experiment and see the changes instantly reflected in the column that lists estimated responses.

- In a Web browser, open the HTML page you created for your experiment. Make sure the browser you use is enabled to read JavaScript.
- Change the value for the factor Front_UCA_Inboard_Height = 100 and Rear_UCA_Inboard_Height = 200 and then select Update.
- If you use the plus or minus buttons to modify values, the responses change dynamically.
- The estimated responses adjust to reflect the new factor values. Notice that the value for both responses variables reflect a change.
- You can continue to vary the factor values and investigate how changes to them affect your responses. To learn more about analyzing the results of your experiment and publishing your results to HTML or SYLK pages, refer to the Adams/Insight online help.

Full Vehicle Analysis (Cont.)

- The results of your experiment may appear as shown below



Responses from Experiment "roll_axis_experiment"

Factor	Current	Min	Nominal	Max
Front_UCA_Inboard_Height	- + 100	2.7190e+01	2.6315e+02	2.7161e+02
Rear_UCA_Inboard_Height	- + 200	3.4013e+01	1.7940e+02	2.8826e+02

Update Reset ☐ Stats ☐ Effects Info...

Response	Current
end_lat_accel	-1.0165
max_yaw_accel	4.1208

- In addition to the basic factor and response information that appears when you first open the HTML file in your Web browser, you can view response statistics and response effects as a function of each factor.

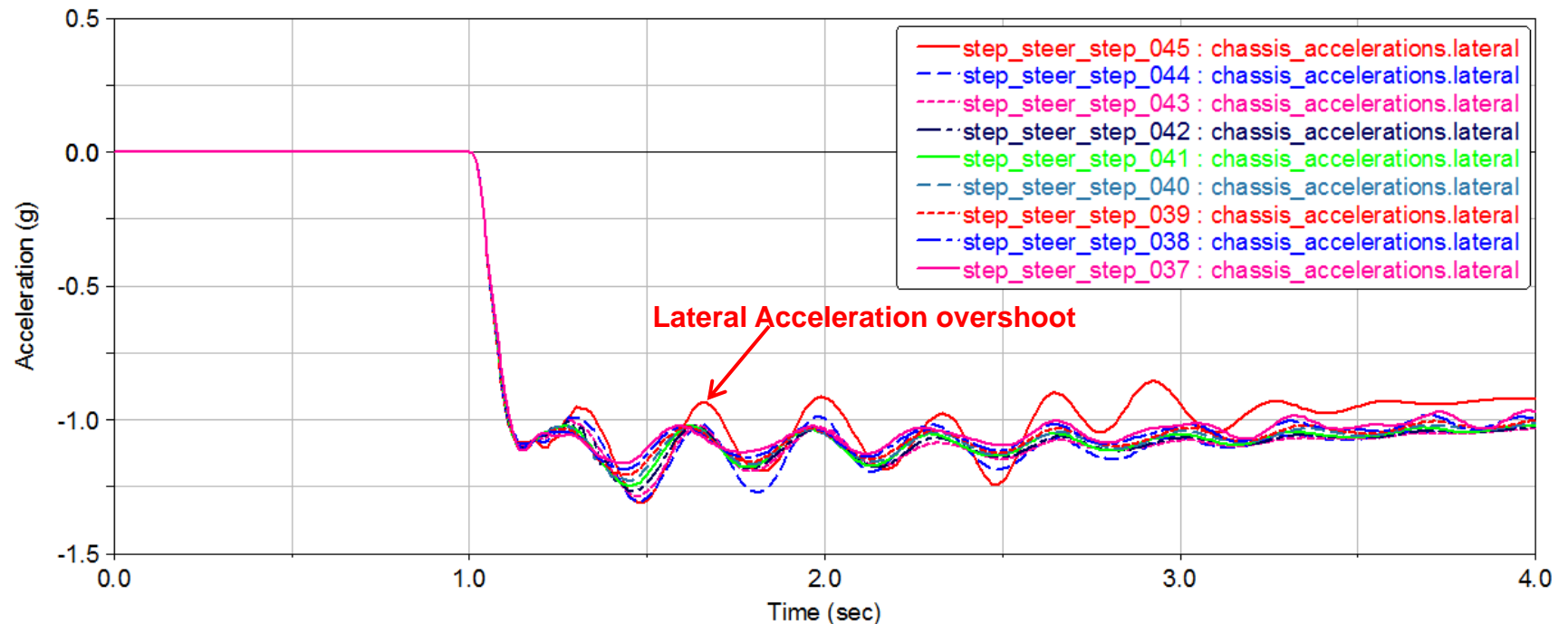
Full Vehicle Analysis (Cont.)

15. Result Interpretation

- For getting good fit we removed outliers so you could not view results above Rear_UCA_Inboard_Height = 288
- However Depending on whether you want better turn-in or maximum lateral acceleration, you would want to pick a rear roll center height near 350mm and a front roll center height between 50mm-100mm.
- Next few slides shows the results for Constant front roll center height of 100mm. Rear roll center height varies from 0mm (step_037) to 400mm (step_045)

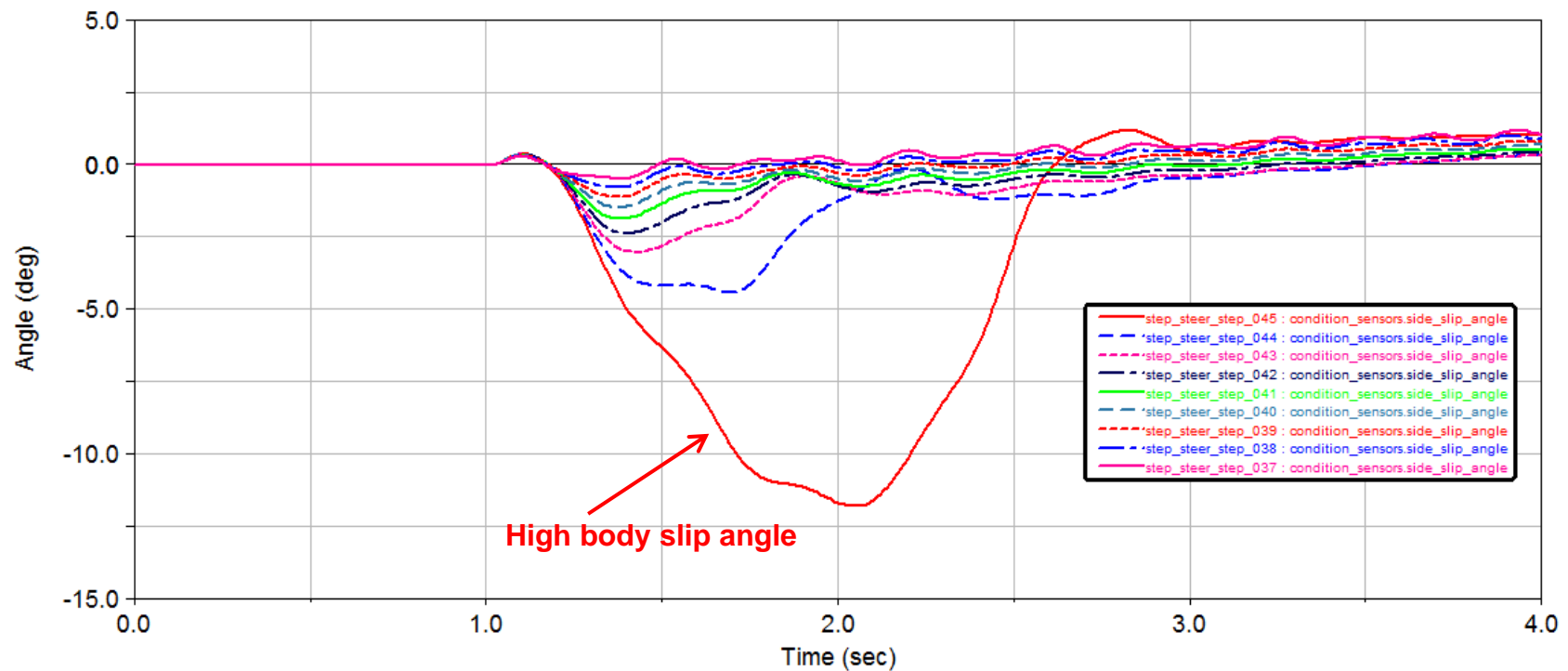
Full Vehicle Analysis (Cont.)

- Chassis Lateral acceleration for Constant front roll center height of 100mm. Rear roll center height varies from 0mm (step_037) to 400mm (step_045).



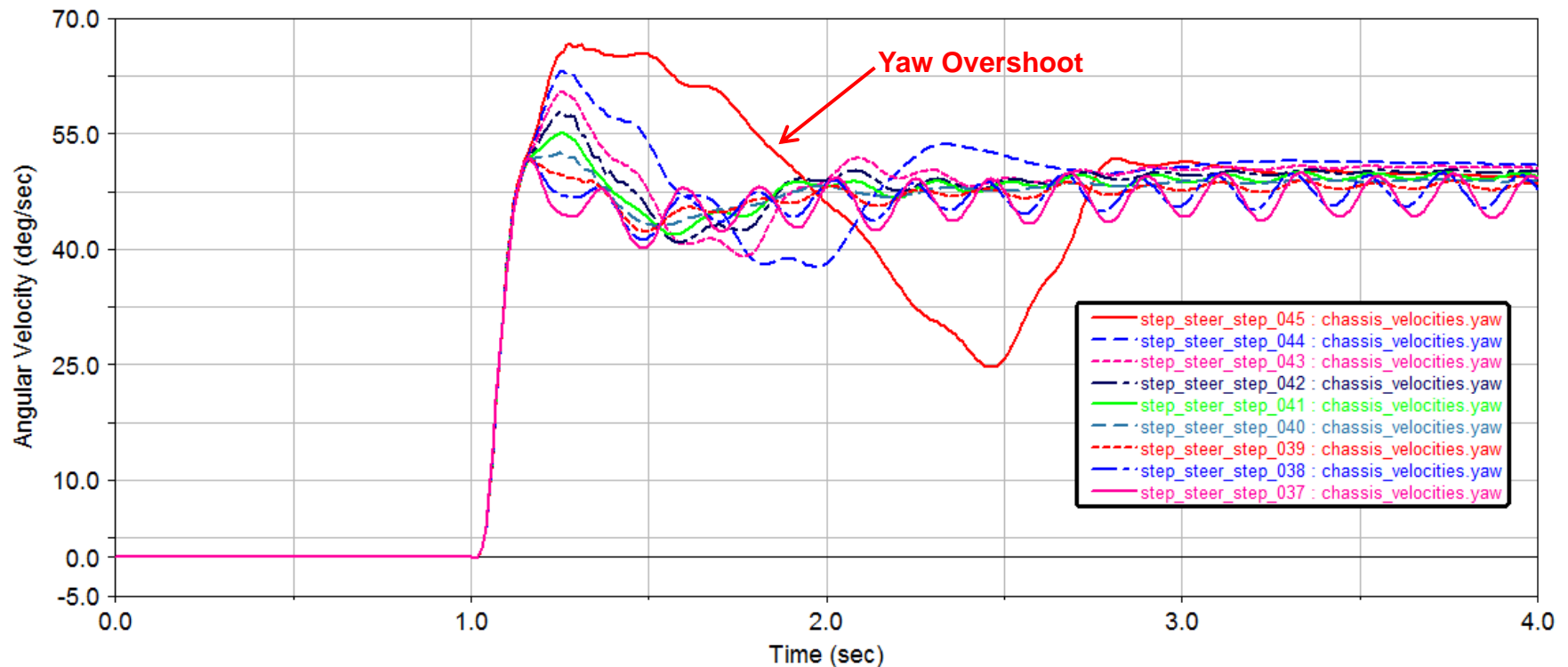
Full Vehicle Analysis (Cont.)

- Side Slip Angle for Constant front roll center height of 100mm. Rear roll center height varies from 0mm (step_037) to 400mm (step_045).



Full Vehicle Analysis (Cont.)

- Chassis Yaw Velocity for Constant front roll center height of 100mm. Rear roll center height varies from 0mm (step_037) to 400mm (step_045).



Conclusion

- **All of these graphs conclude that a nose-down roll axis inclination is ideal for this vehicle setup. But, based on Test 45 where the inclination is most extreme, and poor results are yielded, the roll axis needs to be placed and oriented properly.**
- **Results of test 45**
 - Yaw overshoot is more than 30% and Yaw velocity response time is more; above 1.12 sec
 - Side slip angle is high
 - Lateral acceleration overshoot is above 40% and response time more; above 1 sec

Results for the test 40 looks good so you could run a more detailed D.O.E. to maximize both parameters in this range.

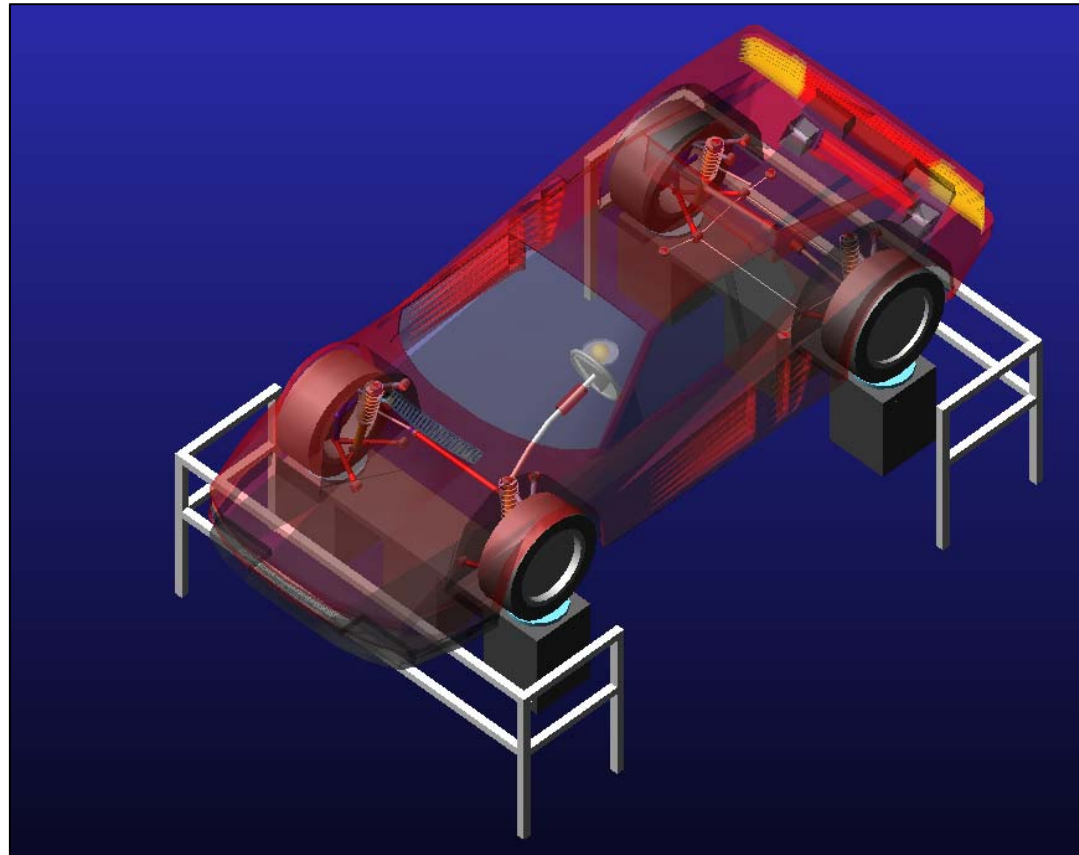
Suggested List of Steps to View and Improve the Fit

- **To Fit Your Results:**

- Here is a suggested list of steps that you can use to view and improve the fit:
- Check R2 and interpret the ANOVA table.
 - Verify residuals
 - Remove outliers, if any
 - Remove terms, if necessary
- Check R2 and interpret the ANOVA table.
 - Transform response, if necessary
 - Change model order, if needed
- Check R2 and interpret the ANOVA table.
- As you attempt these suggestions, go back to through the following steps:
 - Promoting factors
 - Promoting responses
 - Setting design specifications
 - Running your experiment
 - Reviewing the results
 - Fitting results

WORKSHOP 19

SVC & SPMM EVENTS IN ADAMS/CAR



SVC and SPMM Test:

- **Problem Statement:**

- First you will evaluate front and rear suspension characteristics, mass properties and ground reactions using a SVC (Static Vehicle Characteristics) event for both full- and half-vehicle models.
- Next you will perform a SPMM (Suspension Parameter Measurement Machine) analysis to measure Kinematic and Compliance (K&C) characteristics of vehicle. The influence of an antiroll bar in the full vehicle model will be quantified using this analysis.

This workshop contains the following sections:

- To Run a full vehicle SVC
- To Run a half vehicle SVC
- To Run a SPMM test




This workshop takes one hour to complete.

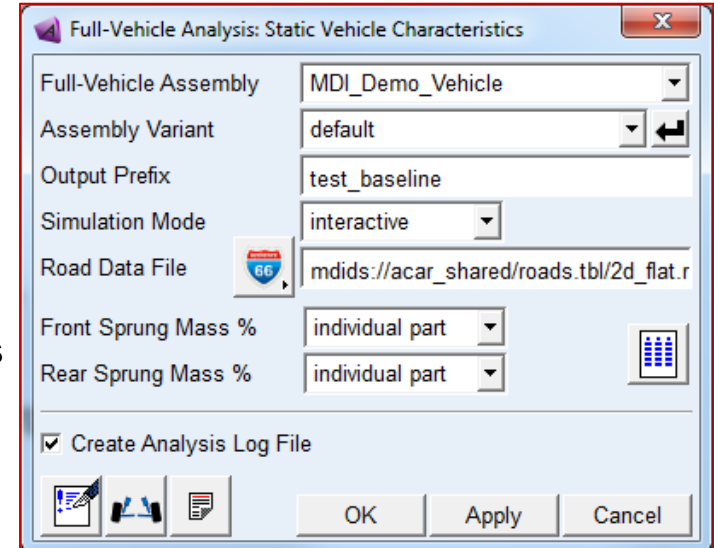
SVC Test 1: Full Vehicle SVC

- **Opening Full Vehicle Assembly :**


1. Start Adams/Car and select **Standard interface** mode.
2. Click **File > Open > Assembly ...**
3. Right click in the assembly name filed, navigate to search then select 'acar_shared'.
4. Open 'MDI_Demo_Vehicle.asy'

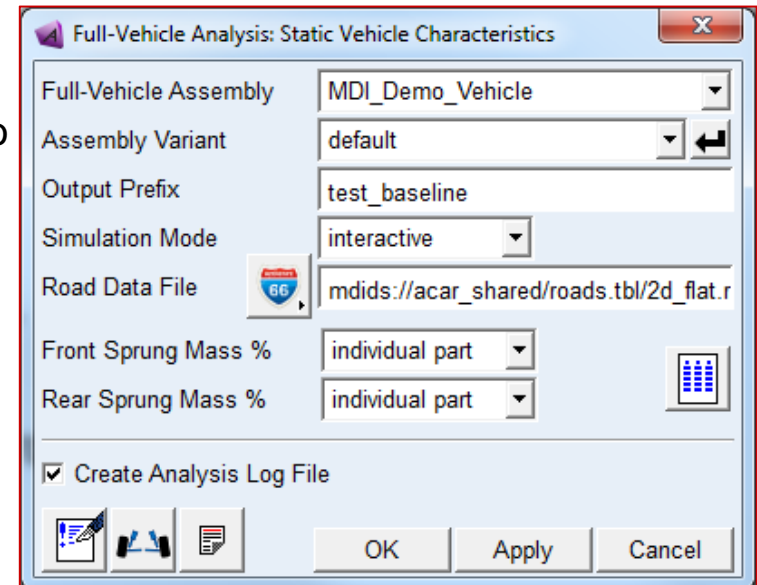
- **Steps for Running Full Vehicle Example:**

1. To perform SVC analysis go to **Simulate > Full Vehicle analysis > Static and Quasi Static Maneuvers > Static Vehicle Characteristics**
2. Enter the output Prefix as **test_baseline**
3. In “**Front Sprung mass %**” and “**Rear Sprung mass %**” Select **Individual part** in options as shown.
4. Click **apply** to run SVC event.
5. SVC Report will be generated automatically. The file generated will be stored in the working directory under name '**test_baseline_svc.svc**'
6. Return to the SVC dialog box and click on data table icon.  This shows the sprung mass table .



SVC Test 1: Full Vehicle SVC

- **Opening Full Vehicle Assembly :**
 1. Start Adams/Car and select Standard interface mode.
 2. Click **File > Open > Assembly...**
 3. Right click in the assembly name filed, navigate to search then select '**acar_shared**'.
 4. Open '**MDI_Demo_Vehicle.asy**'
- **Steps for Running Full Vehicle Example:**
 1. To perform SVC analysis go to **Simulate > Full Vehicle analysis > Static and Quasi Static Maneuvers > Static Vehicle Characteristics**
 2. Enter the output Prefix as **test_baseline**
 3. In "**Front Sprung mass %**" and "**Rear Sprung mass %**" Select **Individual part** in options as shown.
 4. Click **apply** to run SVC event.
 5. SVC Report will be generated automatically. The file generated will be stored in the working directory under name '**test_baseline_svc.svc**'
 6. Return to the SVC dialog box and click on data table icon.  This shows the sprung mass table .



SVC Test 1: Full Vehicle SVC

- Steps for Running Full Vehicle Example continued..
 - From the sprung mass table, change the value of **'gel_lower_control_arm'** to **100**.
 - Click **OK**.
 - Run the SVC again with prefix as **'test'**
 - Select individual part for both **'Front Sprung mass %'** and **'Rear Sprung mass %'**
 - Compare the SVC result files for these analyses and observe the difference in sprung mass calculations.

	Part Name	Mass	% Sprung Mass	SVC Array
1	TR_Front_Suspension.gel_lower_control_arm2	0.1	50.0	front
2	TR_Front_Suspension.gel_tierod_inner	0.3337368666	100.0	front
3	TR_Front_Suspension.gel_tierod_outer	0.3337368666	100.0	front
4	TR_Front_Suspension.gel_lower_control_arm	1.6113954942	100	front
5	TR_Front_Suspension.gel_upright	1.3972982748	0.0	front
6	TR_Front_Suspension.ges_subframe	25.0	100.0	front
7	TR_Front_Suspension.gel_upper_strut	1.0	100.0	front
8	TR_Front_Suspension.gel_upper_control_arm	1.0318710362	50.0	front
9	TR_Front_Suspension.gel_lower_strut	1.0	50.0	front
10	TR_Front_Suspension.gel_spindle	1.1028403931	0.0	front
11	TR_Rear_Suspension.gel_lower_control_arm2	0.1	50.0	rear
12	TR_Rear_Suspension.gel_tierod_inner	0.5	100.0	rear
13	TR_Rear_Suspension.gel_tierod_outer	0.5	0.0	rear
14	TR_Rear_Suspension.gel_lower_control_arm	1.6089138343	50.0	rear
15	TR_Rear_Suspension.gel_upright	1.3294519509	0.0	rear
16	TR_Rear_Suspension.ges_subframe	30.0	100.0	rear
17	TR_Rear_Suspension.gel_upper_strut	5.0	100.0	rear
18	TR_Rear_Suspension.gel_upper_control_arm	1.0318710362	50.0	rear
19	TR_Rear_Suspension.gel_lower_strut	5.0	50.0	rear
20	TR_Rear_Suspension.gel_spindle	1.1028403931	50.0	rear
21	TR_Rear_Suspension.gel_drive_shaft	4.2174529406	50.0	rear

SVC Test 1: Full Vehicle SVC

- Comparison:

Static Vehicle Set-Up Report : test_baseline_svc

```

=====
                        S V C
=====
      STATIC VEHICLE CHARACTERISTICS
      Adams Model Post Processor
=====

```

Adams Model Title: 2015-09-14 17:59:28
<acar_shared>/assemblies.tbl/MDI_Demo_Vehicle.asy

GENERAL CHARACTERISTICS

(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	14.98E+03		
Front ground reaction	N	6304.17	3159.20	3144.97
Rear ground reaction	N	8677.26	4345.82	4331.44
Total roll inertia	kg mm**2	298.7E+06		
Total pitch inertia	kg mm**2	1.162E+09		
Total yaw inertia	kg mm**2	1.340E+09		
Total product Ixy	kg mm**2	1.878E+06		
Total product Ixz	kg mm**2	4.950E+06		
Total product Iyz	kg mm**2	-391.2E+03		
Sprung mass	kg	1396.25		
Sprung roll inertia	kg mm**2	222.9E+06		
Sprung pitch inertia	kg mm**2	941.7E+06		
Sprung yaw inertia	kg mm**2	1.051E+09		
Sprung product Ixy	kg mm**2	1.890E+06		
Sprung product Ixz	kg mm**2	1.418E+06		
Sprung product Iyz	kg mm**2	-356.7E+03		
Total c.g. height	mm	401.63		
Sprung c.g. height	mm	409.26		
Body yaw angle	DEG	0.00		
Body pitch angle	DEG	-540.8E-03		
Body roll angle	DEG	16.34E-03		

Close

Static Vehicle Set-Up Report : test_svc

```

=====
                        S V C
=====
      STATIC VEHICLE CHARACTERISTICS
      Adams Model Post Processor
=====

```

Adams Model Title: 2015-09-14 18:01:25
<acar_shared>/assemblies.tbl/MDI_Demo_Vehicle.asy

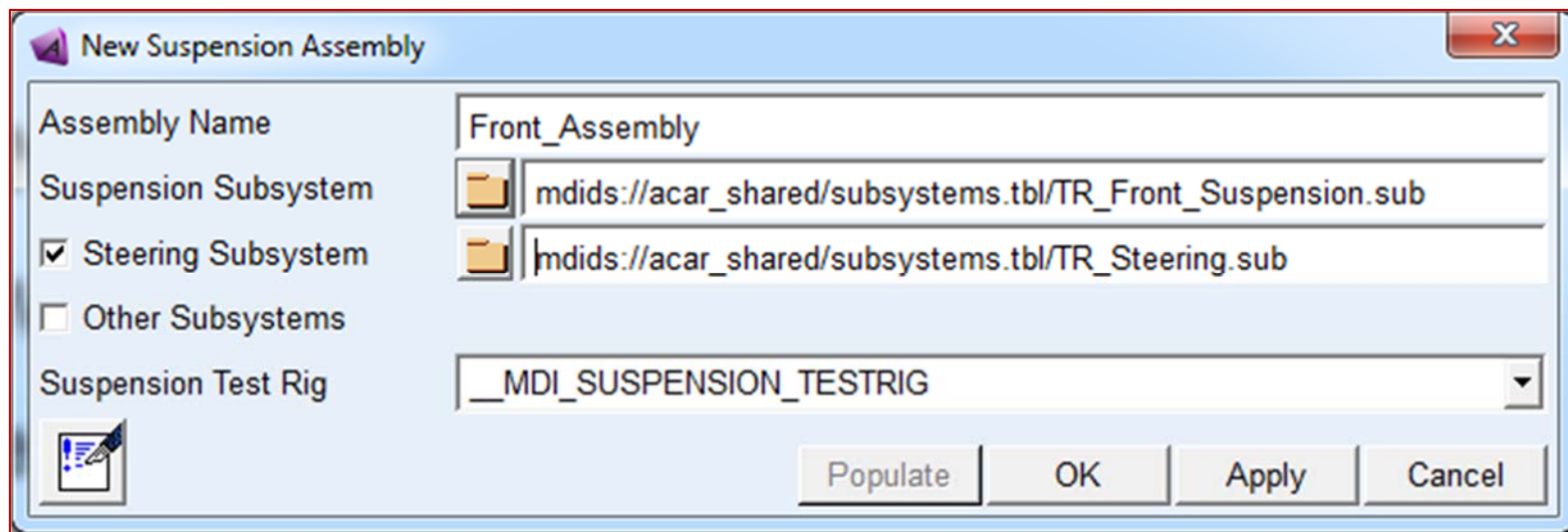
GENERAL CHARACTERISTICS

(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	14.98E+03		
Front ground reaction	N	6304.17	3159.20	3144.97
Rear ground reaction	N	8677.26	4345.82	4331.44
Total roll inertia	kg mm**2	298.7E+06		
Total pitch inertia	kg mm**2	1.162E+09		
Total yaw inertia	kg mm**2	1.340E+09		
Total product Ixy	kg mm**2	1.878E+06		
Total product Ixz	kg mm**2	4.950E+06		
Total product Iyz	kg mm**2	-391.2E+03		
Sprung mass	kg	1397.86		
Sprung roll inertia	kg mm**2	223.6E+06		
Sprung pitch inertia	kg mm**2	945.4E+06		
Sprung yaw inertia	kg mm**2	1.055E+09		
Sprung product Ixy	kg mm**2	1.886E+06		
Sprung product Ixz	kg mm**2	2.094E+06		
Sprung product Iyz	kg mm**2	-357.5E+03		
Total c.g. height	mm	401.63		
Sprung c.g. height	mm	408.95		
Body yaw angle	DEG	0.00		
Body pitch angle	DEG	-540.8E-03		
Body roll angle	DEG	16.34E-03		

Close

SVC Test 1: Full Vehicle SVC

- **Steps for opening Half Vehicle Assembly :**
 1. Start **Adams/Car** and select **Standard interface** mode.
 2. Click **File > New > Suspension assembly...**
 3. Fill out the box as shown.



4. Click **OK**.

SVC Test 2: Full Vehicle SVC

- **Steps for Setting Suspension Parameters:**

1. To set parameters go to **Simulate > Suspension Analysis > Set** Suspension parameters.
2. Use the sprung mass, Unsprung CG height [Front suspension characteristics] and wheel base value from report of full vehicle SVC analysis. (open .SVC file with notepad).
3. Similarly, note the values of Front ground reaction, Sprung mass, wheelbase, Unsprung cg height, wheel center rise and static loaded tire radius from full vehicle SVC file.(note: edit **.svc file** with **notepad**). For help see the next slide.
4. Click **OK** .

Suspension Analysis: Setup Parameters

Suspension Assembly: Front_Assembly

Suspension Settings

Tire Model: Property File

Tire Unloaded Radius: 300.0

Tire Stiffness: 200.0

Tire Property File: mdids://acar_shared/tires.tbl/TR_front_pac89.tir

Wheel Mass: 1.0

Dual Wheels: ☐ Active ☒ Inactive

Dual Wheel Offset: 300.0

Vehicle Parameters

Sprung Mass: 1396.25

CG Height: 316.06

Wheelbase: 2599.78

Drive Ratio (% Front): 50

Brake Ratio (% Front): 55

OK Apply Cancel

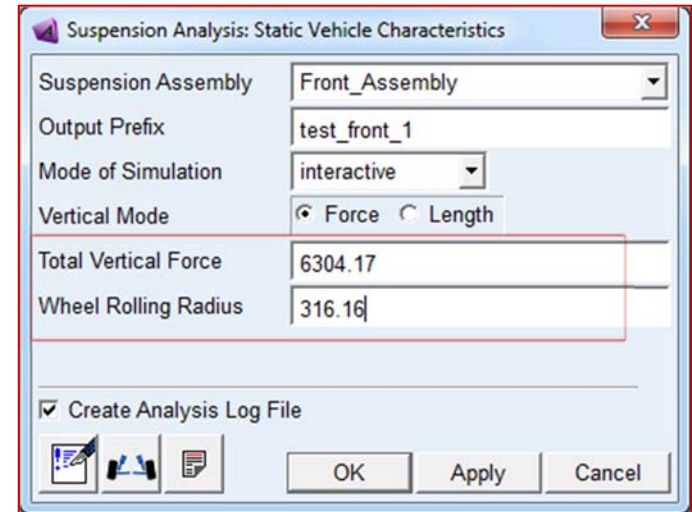
SVC Test 2: Full Vehicle SVC

- svc file for full vehicle body

GENERAL CHARACTERISTICS				
(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	14.98E+03		
Front ground reaction	N	6304.17	3159.20	3144.97
Rear ground reaction	N	8677.26	4345.82	4331.44
Total roll inertia	kg mm**2	298.7E+06		
Total pitch inertia	kg mm**2	1.162E+09		
Total yaw inertia	kg mm**2	1.340E+09		
Total product Ixy	kg mm**2	1.878E+06		
Total product Ixz	kg mm**2	4.950E+06		
Total product Iyz	kg mm**2	-391.2E+03		
Sprung mass	kg	1396.25		
Sprung roll inertia	kg mm**2	222.9E+06		
Sprung pitch inertia	kg mm**2	941.7E+06		
Sprung yaw inertia	kg mm**2	1.051E+09		
Sprung product Ixy	kg mm**2	1.890E+06		
Sprung product Ixz	kg mm**2	1.418E+06		
Sprung product Iyz	kg mm**2	-356.7E+03		
Total c.g. height	mm	401.63		
Sprung c.g. height	mm	409.26		
Body yaw angle	DEG	0.00		
Body pitch angle	DEG	-540.8E-03		
Body roll angle	DEG	16.34E-03		
Speed	mm/S	0.00		
wheelbase	mm	2559.78	2559.79	2559.78
(PARAMETER)	(UNITS)	(AVERAGE)	(LEFT)	(RIGHT)
Unsprung mass (total)	kg	58.74		
Unsprung c.g. height	mm	315.50		
Roll center height	mm	149.55		
wheel center rise	mm	-1.01	-1.20	-812.4E-03
Static loaded tire radius	mm	316.06	316.03	316.08
Track width	mm	1526.34		
Axle distance from vehicle cg	mm	1482.68		

SVC Test 2: Full Vehicle SVC

- Steps for running SVC on a half Vehicle Assembly
- To perform SVC go to Simulate > Suspension analysis > Static Vehicle Characteristics.
 1. Fill the dialog box based on inputs from full vehicle SVC. (Inputs: Ground reaction force and tire radius)
 2. Click 'Apply'. (Don't Close the Dialog box we need it for further example).
 3. Report will be generated and stored in working directory automatically 'Test_front_1_susp_svc.svc' .
 4. Compare the results with full vehicle analysis.
- We can observe that most of the results match approximately. This difference in results is due to the average value for total vertical force we applied.



Suspension Analysis: Static Vehicle Characteristics

Suspension Assembly: Front_Assembly

Output Prefix: test_front_1

Mode of Simulation: interactive

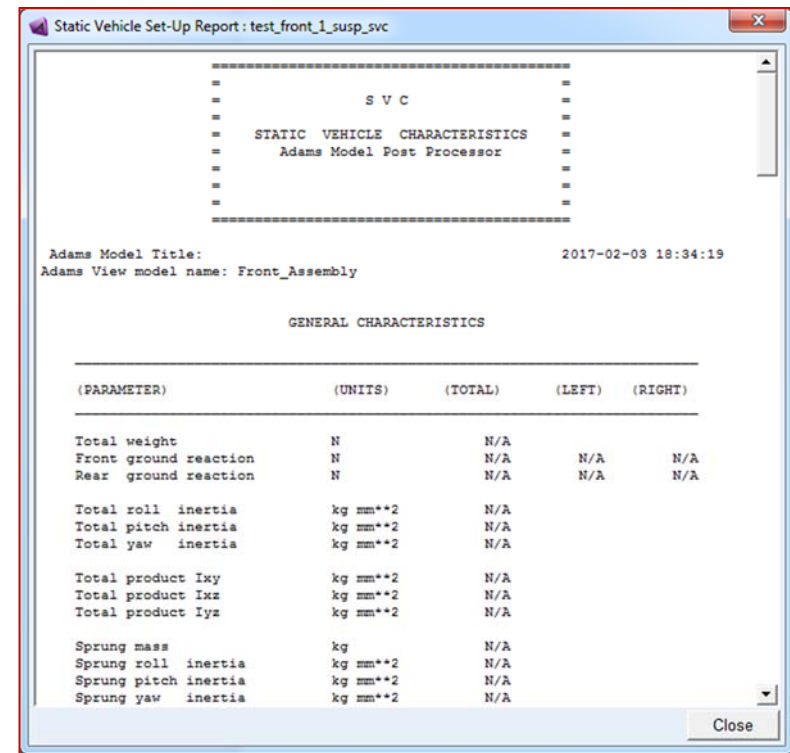
Vertical Mode: ☒ Force ☐ Length

Total Vertical Force: 6304.17

Wheel Rolling Radius: 316.16

☒ Create Analysis Log File

OK Apply Cancel



Static Vehicle Set-Up Report : test_front_1_susp_svc

S V C

STATIC VEHICLE CHARACTERISTICS

Adams Model Post Processor

Adams Model Title: 2017-02-03 18:34:19

Adams View model name: Front_Assembly

GENERAL CHARACTERISTICS

(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	N/A		
Front ground reaction	N	N/A	N/A	N/A
Rear ground reaction	N	N/A	N/A	N/A
Total roll inertia	kg mm**2	N/A		
Total pitch inertia	kg mm**2	N/A		
Total yaw inertia	kg mm**2	N/A		
Total product Ixy	kg mm**2	N/A		
Total product Ixz	kg mm**2	N/A		
Total product Iyz	kg mm**2	N/A		
Sprung mass	kg	N/A		
Sprung roll inertia	kg mm**2	N/A		
Sprung pitch inertia	kg mm**2	N/A		
Sprung yaw inertia	kg mm**2	N/A		

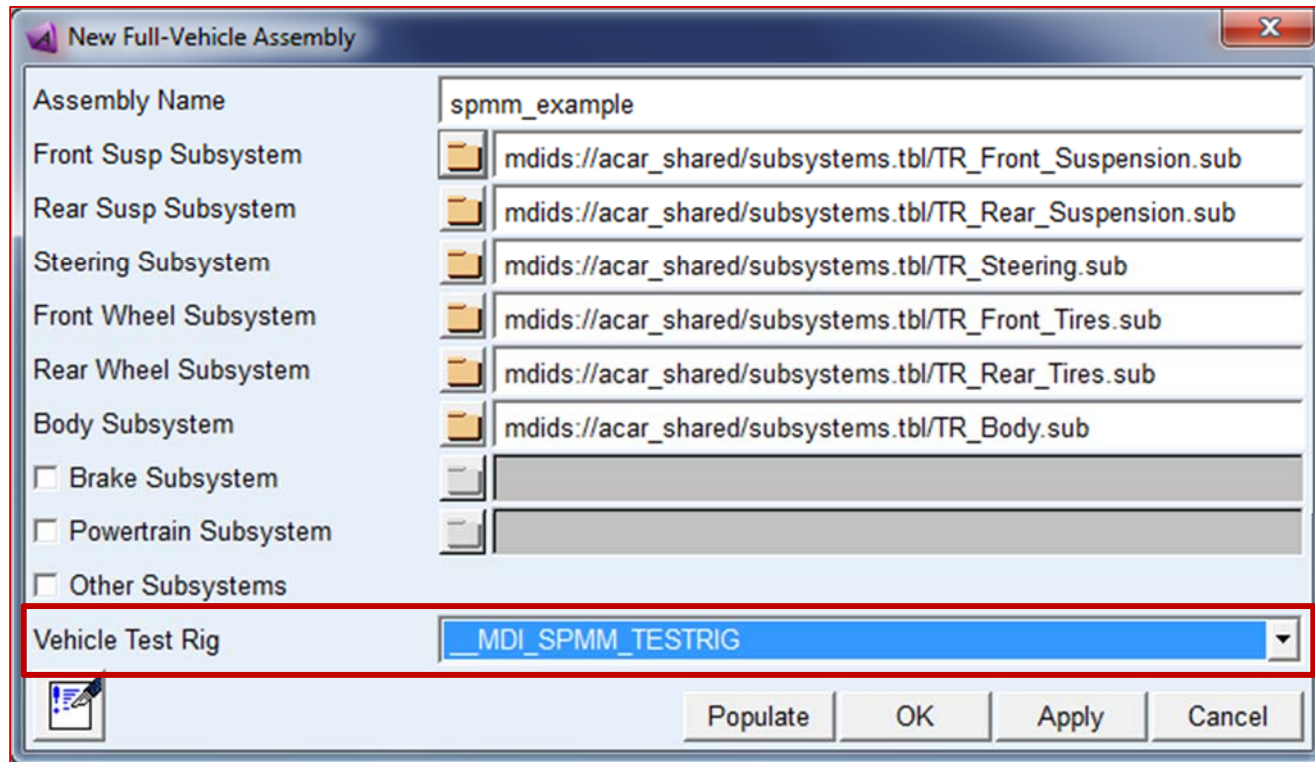
Close

Running SPMM

- **Opening the model**

- Creating a new assembly using the SPMM TESTRIG :

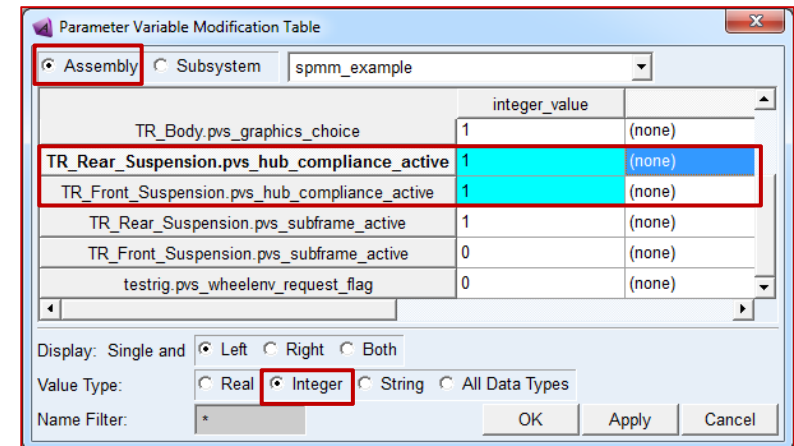
1. Start Adams/Car and select **Standard Interface mode**.
2. Click **File > New > Full-Vehicle Assembly...**
3. Fill out the box as shown below and click **OK**.



Running SPMM

4. Activate the adjuster forces in the front and rear suspension subsystems so that you can specify alignment (toe and camber) setting in the SPMM event dialog box .

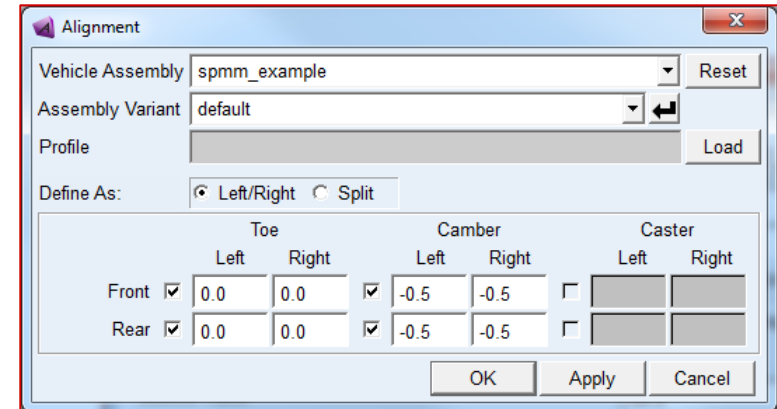
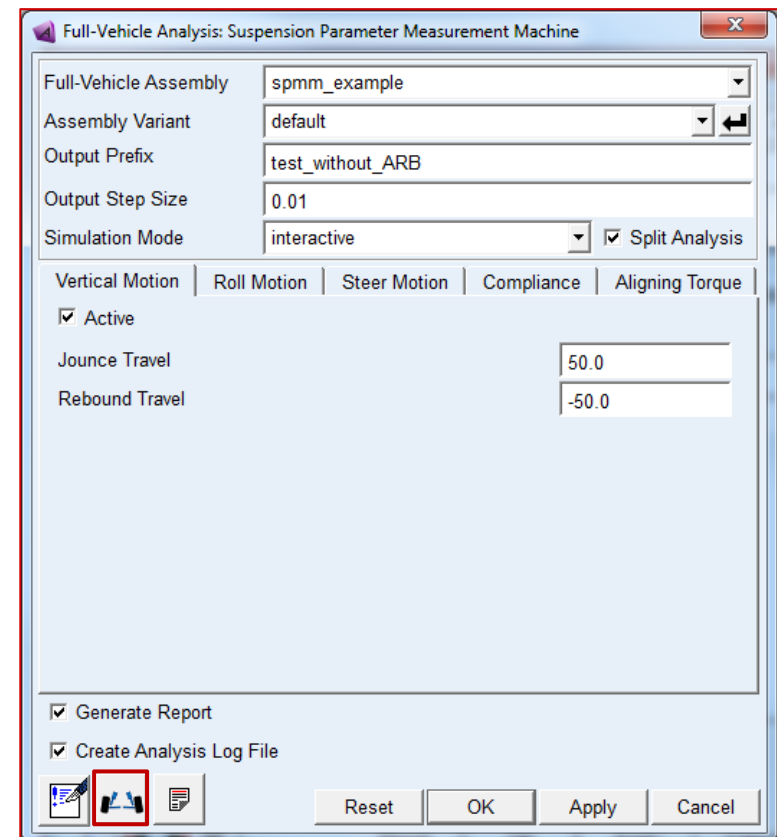
- Go to **Adjust > Parameter Variable > Table**
- Select **Assembly** and value type as Integer
- Set both **TR_Front_Suspension.pvs_hub_compliance_active** and **TR_Rear_Suspension.pvs_hub_compliance_active** to 1 as shown in below image



5. Save the assembly in the acar_training_MD database
 - **File > Save As > Select target database as acar_training_MD > click OK**

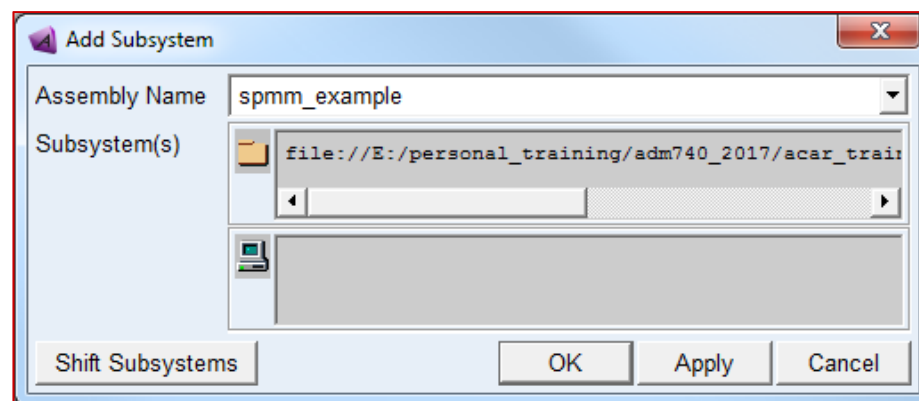
Running SPMM

- **Perform SPMM Analysis :**
 1. Select **Full-Vehicle Analysis > Kinematic and Compliance > Suspension Parameter Measurement Machine...**
 2. Enter the **output prefix** as **test_without_ARB**
 3. We will use default values for now for all the sub events, select **OK** to run SPMM event
 4. SPMM Report will be visible on your screen once the simulation completed . This reports the information at static condition related to vehicle K&C characteristics.
- The sign convention used for SPMM reporting is documented at the end of the report.



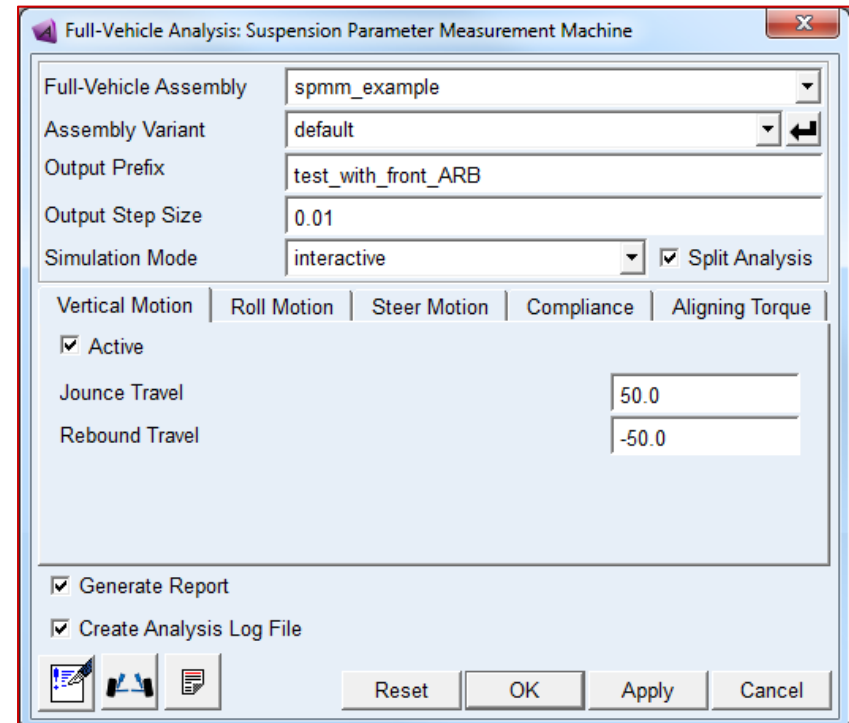
Running SPMM – with ARB incorporated

- **Re-run with model containing Anti-Roll Bar (ARB):**
 1. We will use the subsystem **MDI_ANTI_ROLL_FRONT.sub** provided in **acar_training_MD.cdb** database.
 2. Add this **MDI_ANTI_ROLL_FRONT.sub** to the full vehicle assembly using option **File > manage > assemblies > Add subsystem** like shown in image here
 3. Right click in the subsystem tab and browse the **MDI_ANTI_ROLL_FRONT.sub** from the **acar_training_MD.cdb**
 4. Save Assembly in **acar_training_MD.cdb** database as **spmm_example_front_ARB**
File > Save As ,select database as **acar_training_MD.cdb** and click **OK**.



Running SPMM – with ARB incorporated

- Perform an SPMM Analysis with ARB :
 1. Perform a SPMM analysis, by clicking Simulate then navigating to **Full Vehicle Analysis > Kinematic and Compliance > Suspension Parameter Measurement Machine...**
 2. Use default values for now, select **OK** to run SPMM event
 3. SPMM Report will be visible on your screen once the simulation completed .



Full-Vehicle Analysis: Suspension Parameter Measurement Machine

Full-Vehicle Assembly:

Assembly Variant:

Output Prefix:

Output Step Size:

Simulation Mode: ☒ Split Analysis

Vertical Motion | Roll Motion | Steer Motion | Compliance | Aligning Torque

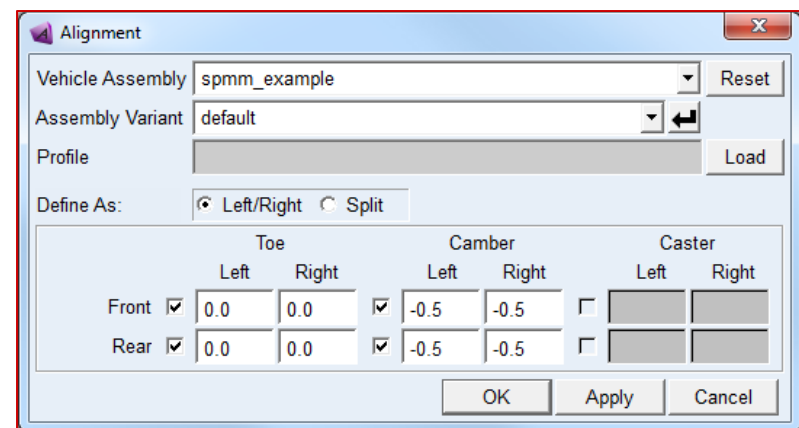
☒ Active

Jounce Travel:

Rebound Travel:

☒ Generate Report

☒ Create Analysis Log File



Alignment

Vehicle Assembly:

Assembly Variant:

Profile:

Define As: ☒ Left/Right ☐ Split

		Toe			Camber			Caster	
		Left	Right		Left	Right		Left	Right
Front	<input checked="" type="checkbox"/>	<input type="text" value="0.0"/>	<input type="text" value="0.0"/>	<input checked="" type="checkbox"/>	<input type="text" value="-0.5"/>	<input type="text" value="-0.5"/>	<input type="checkbox"/>	<input type="text" value=""/>	<input type="text" value=""/>
Rear	<input checked="" type="checkbox"/>	<input type="text" value="0.0"/>	<input type="text" value="0.0"/>	<input checked="" type="checkbox"/>	<input type="text" value="-0.5"/>	<input type="text" value="-0.5"/>	<input type="checkbox"/>	<input type="text" value=""/>	<input type="text" value=""/>

Comparison of SPMM Results: Effect of ARB

- Compare the results for Roll parameters for the front axle side for the Roll test. Notice the change in the roll parameters when using an ARB.

test_without_arb_spm_rep.dat

ROLL TEST:				
Wheel rate in roll	N/mm	36.49	36.46	36.52
Roll stiffness	Nmm/deg	7.41e+005		
Roll steer (per degree)	deg/deg	0.44231	0.44868	0.43595
Roll steer	deg/mm	-0.00017	-0.03719	0.03684
Roll camber w.r.t. chassis (per degree)	deg/deg	-0.22670	-0.22490	-0.22850
Roll camber (per degree)	deg/deg	0.76993	0.77169	0.76818
Roll camber	deg/mm	0.00007	-0.01904	0.01919

test_with_ARB_spm_rep.dat

ROLL TEST:				
Wheel rate in roll	N/mm	62.03	61.88	62.17
Roll stiffness	Nmm/deg	1.26e+006		
Roll steer (per degree)	deg/deg	0.43264	0.43913	0.42616
Roll steer	deg/mm	-0.00015	-0.03892	0.03862
Roll camber w.r.t. chassis (per degree)	deg/deg	-0.23897	-0.23806	-0.23989
Roll camber (per degree)	deg/deg	0.75763	0.75849	0.75676
Roll camber	deg/mm	0.00007	-0.02155	0.02169

APPENDIX A

TYPICAL WORKFLOW WITH ADAMS/CAR

Example Analyses

- This appendix provides a basic outline of how you can use Adams/Car to analyze your vehicle.



This workshop takes one hour to complete.

Example Analyses

- **What's in this appendix:**
 - Types of Analyses
 - Gather Data for the Model
 - Packaging Analysis on Suspension
 - Kinematic Analysis on Suspension
 - Suspension-Compliance Analysis
 - Static-Loading Durability Analysis
 - Dynamic-Loading Durability Analysis
 - Front Suspension Analyses
 - Full-Vehicle Design and Analysis

Types of Analyses

- **The following is a list of analyses that you can perform on Adams/Car:**
 - Packaging analysis on suspension
 - Kinematic analysis on suspension
 - Suspension-compliance analysis on suspension
 - Static-loading durability analysis on suspension
 - Dynamic-loading durability analysis on suspension
 - Full-vehicle design and analysis
- **In the following pages, a guide is given to perform these analyses. This guide only lists a subset of the analyses available in Adams/Car. Moreover, some additional modules may be necessary for the analyses listed (for example, Adams/Durability).**



For information on a particular analysis, see the Adams/Car online help.

Gather Data for the Model

- **Collect the data necessary to generate an Adams model of the proposed suspension concept. After you've collected the data from various sources (CAD data for geometry, bushing data from internal testing facilities or from bushing supplier, shock data, and so on), you can create an Adams/Car model.**

Packaging Analysis on Suspension

- Once you've created the Adams model, you put the virtual suspension on a virtual test fixture (standard part of Adams/Car) and run through a series of events to examine packaging and interference issues. The goal of this analysis is to show that parts do not collide during jounce and roll travel.
- Also, if body geometry is available, be able to demonstrate that tire and wheel well clearances conform to corporate standards. The goal of this phase of the analysis is to give a cursory check of the part collisions within the Adams/Car environment.
- Further investigations are possible by bringing the Adams results for the wheel envelope during maximum jounce/rebound/roll travel into your CAD package. You can use the solid geometry of the wheel envelope in your CAD package to easily find clearance issues, and to better visualize the total wheel envelope needed by your suspension concept.

Kinematic Analysis on Suspension

- **After you've analyzed the suspension packaging issues, put the virtual suspension on a virtual test fixture and run through a series of events to understand the kinematic properties of the suspension. Two analyses help you understand the suspension kinematics: parallel wheel travel (wheels moving vertically in phase) and roll travel (wheel moving vertically out of phase).**
- 1. The parallel wheel travel analysis (also called the ride-travel event) examines the following suspension metrics:**
 - Toe (also called bump steer)
 - Caster
 - Camber
 - Longitudinal recession
 - Lateral recession
 - Wheel rate (vertical force versus amount of suspension vertical deflection)

Kinematic Analysis on Suspension

2. The roll-travel analysis examines the following suspension metrics:

- Roll steer (degrees of toe per degree of suspension roll)
- Roll stiffness
- The goal of the kinematic analysis is to tune the geometry of the suspension to attain satisfactory kinematic behavior. If kinematic issues arise, design recommendations can then be made to the location of suspension joints and bushings, the lengths of the control arms, and other geometric properties that affect the kinematics of the suspension.
- Also, suspension spring properties can be examined to be sure that the overall vehicle requirements will be met for suspension springs during ride and roll. Use of additional suspension components such as an anti-roll bar can be examined, including recommendations about the sizing for the anti-roll bar.

Suspension-Compliance Analysis

- After you analyze the suspension geometry and the design shows good kinematic behavior, you can examine suspension compliance. The virtual suspension model will be placed on the suspension test rig and run through the compliance analysis (for example, static loading). The following metrics are generated with this analysis:
 - Lateral force versus toe (for both parallel and opposing lateral force)
 - Lateral force versus camber (for both parallel and opposing lateral force)
 - Lateral force versus lateral displacement (for both parallel and opposing lateral force)
 - Longitudinal force versus toe (braking and acceleration forces)
 - Longitudinal force versus longitudinal displacement (braking and acceleration forces)
 - Aligning torque versus toe (parallel and opposing torques)

Suspension-Compliance Analysis

- The goal of the suspension compliance analysis is to tune the suspension bushings such that adequate suspension compliance is attained. Note that real joints are not infinitely stiff, and they do factor in to the suspension performance. Thus, it may be a good idea to replace idealized joints in your model with bushing representations.

Static-Loading Durability Analysis

- The next step in analyzing the suspension is to apply static loadcases to the wheels in Adams/Car and examine the resulting loads on the suspension elements (suspension bushings, suspension springs, and so on). This contributes to a better understanding of the durability of the suspension.
- Typically, a set of requirements for static loads are used which take the form of a worst case loading condition that a suspension must withstand. These loading conditions (or loadcases) take the form of number of g's of loading. For example, a suspension requirement might be that it must withstand 3 g's of vertical load, 2 g's of longitudinal load, and 1 g of lateral load. Such a loading condition is often referred to as a "321 g" loadcase.
- The use of "g's" describes the total vehicle weight, divided by the front to rear load distribution. In this way, a loadcase requirement in "g's" of load can be used across different vehicles of various sizes and weights.

Static-Loading Durability Analysis

- The goal of the loadcase analysis is to give the design and FEA analyst a report which shows the worst case loading on each of the suspension components (control arm, bushings, spring, and so on). This data can then be fed into a FEA model of the component (bushing, control arm, spring, and so on) to show that the component will withstand a given amount of static loading. A structured report will be generated through the use of the loadcase postprocessor which will show the amount of loading on the suspension elements, and using this report the loading data can be imported into the FEA software (NASTRAN, ANSYS, ABAQUS, and so on).
- It is rare that the design engineer can have access to this level of data early in the program, so the static loadcase analysis provides the first insight to real-world loading conditions for the parts. This method is coarse and does not provide the final answer to the durability requirements of the suspension components, but early in the design and analysis, this information is extremely valuable.

Dynamic-Loading Durability Analysis

- **After the static durability analysis is completed, a more intensive investigation into the rubber mounts begins. In this phase, a road profile will be assumed for the suspension (for example, a pothole, or random white noise road input) and utilized on a virtual four-post shaker event within Adams/Car. Adams dynamic loading histories of the mounts will then be imported into FEA software for final structural analysis.**

Front Suspension Analyses

- Besides the analyses already listed, specific to the front suspension, steering system metrics will also be examined to understand the overall steering ratio, steering linearity, Ackerman, and steering column universal joint phasing (ensure that the steering feel is not lumpy).

Full-Vehicle Design and Analysis

- **Once the rear suspension, front suspension, and steering system subsystems have been modeled in Adams/Car, the next stage of the Functional Digital Car process is to analyze the full vehicle behavior. Because the various suspension subsystem models have been developed in the earlier stages of the plan, it will be a simple matter to assemble these subsystems into a full vehicle Adams/Car model.**
- **This Adams/Car model will be driven on a set of virtual test tracks to understand various full vehicle metrics. Example test analyses would be:**
 - Static vehicle trim analysis to trim the vehicle to a particular ride height

Full-Vehicle Design and Analysis

- Double lane change (Moose Test) to examine rollover propensity in an accident-avoidance maneuver
- Constant radius cornering to examine understeer/oversteer behavior of the vehicle and generate an understeer budget report
- Brake drift to examine amount of lateral drift during a moderate-braking maneuver
- Brake in a turn to examine dynamic braking and turning stability issues
- Step steer to examine lateral acceleration and yaw rate overshoot during highly dynamic maneuver
- Frequency response to examine vehicle dynamics metrics in the frequency domain
- Dynamic durability to determine the behavior of the vehicle as it drives over potholes, bumps and other three dimensional road obstacles



APPENDIX B

ADAMS/CAR FILES

Adams/Car Files

- This appendix supplies information about the Adams/Car configuration and data files.

Adams/Car Files

- **What's in this appendix:**
 - Adams/Car Configuration Files
 - Adams/Car Data Files

Adams/Car Files

Table 1. Adams/Car Configuration Files

The file:	Contains:
.acar.cfg	Information that Adams/Car reads during startup to correctly initialize the session. There are private, shared, and site configuration files.
acar.cmd	Commands for starting Adams/Car.
acarAS.cmd	Preferences you set. AS stands for <i>After Startup</i> , meaning that Adams/Car reads it after it reads other setup files.
acarBS.cmd	Preferences you set. BS stands for <i>Before Startup</i> , meaning that Adams/Car reads it before it reads other setup files.

Adams/Car Files

Table 2. Adams/Car Data Files	
The File:	Does the following:
Aero_force(.aer)	Contains wind-force mappings.
Assembly(.asy)	Lists the subsystems that make up Adams/Car assemblies.
Adams/Car database (.cdb)	Directory that serves as the Adams/Car database.
Drive control (.dcf)	Contains maneuver descriptions for the Driving Machine.
Driver data (.dcd)	Contains data for the Driving Machine.
Differential (.dif)	Defines the slip speed-torque characteristics of a differential.
Driver loadcase (.dri)	Contains driving signals used in a data-driven, full-vehicle analysis. The driver loadcase specifies inputs to the vehicle.
Loadcase (.lcf)	Contains data used in suspension analyses.

Adams/Car Files

Table 2. Adams/Car Data Files (continued)	
The file:	Does the following
Model (.mdl)	Obsolete
Plot configuration (.plt)	Defines a suit of plots to be automatically generated after completion of an analysis.
Powertrain (.pwr)	Defines the engine speed-torque relationship at different throttle positions.
Property	Contains force properties of entities: <ul style="list-style-type: none"> " Bump stops (.bum), .xml " Bushings (.bus), .xml " Dampers (.dpr), .xml " Reboundstop (.reb), .xml " Spring (.spr), .xml " Tire (.tir)
Road Data (.rdf)	Contains data on road.
Subsystem (.sub)	Contains information unique to the specific instance of the template the subsystem file references.
Suspension curves (.scf)	Used in Conceptual Suspension Modeling add-on module.
Steering_assist (.ste)	Contains torsion bar data relating torsion bar deflection to both torque and pressure.
Tables (.tbl)	Subdirectory in Adams/Car database called tables. Each subdirectory contains files for specific types of components, such as springs and dampers, or files for performing tests, such as loadcases and wheel envelopes.
Template file (.tpl)	Defines the topology and major role (for example, suspension or steering) of Adams/Car models
Wheel envelope (.wen)	Contains location vector information that represents the wheel center location and orientation for space. Used for wheel envelope analyses.

APPENDIX C

STRUCTURE OF EVENT FILE AND DRIVER CONTROL DATA

Specifying End Conditions

- **EXPERIMENT:** In the **EXPERIMENT** section,
 - Specifies the static setup, initial speed, initial gear, and a list of mini-maneuvers for the experiment.
 - The Driving Machine executes each mini-maneuver in the order listed, until the list is complete or a mini-maneuver is terminated.
 - you must supply parameters for INITIAL_SPEED, and optionally INITIAL_GEAR (default is 3) and INITIAL_CLUTCH (default is 0, engaged).
 - Optionally, you can specify static-setup analyses that remove start-up transients, which can eliminate mini-maneuvers that you might normally use when setting up a vehicle for cornering maneuvers.
 - You may choose any of the values below for STATIC_SETUP. If you omit the STATIC_SETUP parameter from your driver control file, Adams/Car uses a default value of NORMAL.
 - NORMAL
 - NONE
 - SETTLE
 - STRAIGHT
 - SKID_PAD



For information see in the Adams/Car online help or press the F1 key

Specifying End Conditions

- **MINI MANEUVER:**

- The EXPERIMENT section is followed by a MINI_MANEUVER section.
- For each mini-maneuver, the MINI_MANEUVER section must specify a name for the mini-maneuver, the abort time, and the output step size.
- Optionally, you can define the condition for ending the mini-maneuver, and the maximum integration step size, hMax (which must be smaller than the output step size).
- If the abort time is reached before the end condition is satisfied, the Driving Machine terminates the whole experiment, not just the mini-maneuver. You can also specify an end-condition subblock which will apply to all mini-maneuvers in the experiment.
- The following shows an example:

Name	Active	Abort Time	Step Size	Sample Period
BRAKE_TURN	yes	10.0	0.01	0.01

Add Name Type: DcfMini Parent: maneuver

Specifying End Conditions

- An event file can contain an unlimited number of mini-maneuver sections.
- Each mini-maneuver section contains sections titled (STEERING), (THROTTLE), (BRAKE), (GEAR) and (CLUTCH).
- Depending on the control methods you specify, you may have to choose a (MACHINE_CONTROL) or (OPEN_CONTROL) block.
- A mini-maneuver may also include an (END_CONDITIONS) data, where a condition may be specified which will trigger the passing of control to the next mini-maneuver. In general, such a block is required for all but the final mini-maneuver, to ensure that all mini-maneuvers are executed before the abort time is reached.
- The following shows an example mini-maneuver section, for a mini-maneuver named BRAKE_TURN:

The screenshot shows a software interface for configuring a mini-maneuver named "BRAKE_TURN". The window has a title bar with "Name" and "Comment" fields. Below the title bar are tabs for "Steering", "Throttle", "Braking", "Gear", "Clutch", and "Conditions". The "Steering" tab is selected. Inside the "Steering" tab, there are several settings: "Actuator Type" is set to "rotation", "Control Method" is set to "machine", and "Control Type" is set to "constant". There are also radio buttons for "Control Mode" with "Absolute" and "Relative" options. To the right, there is a "Steering" section with a "Steer Control" dropdown set to "skidpad", an "Entry Distance" field set to "0.0", and a "Radius" field set to "20.0". Below these is a "Turn Direction" section with radio buttons for "Left" and "Right". At the bottom of the window, there is a "Current Field Unit" field set to "length (meter)" and four buttons: "Save & Run", "Save", "Save As", and "Cancel".

Specifying End Conditions

- You can specify end conditions for each mini-maneuver in a sub-block. The **END_CONDITIONS** sub-block looks like this:

Name	Type	Condition Sensor	Test	Trigger Value	Error	Filter Time	Delay Time	Group Name
END_1	lat_accel		==	2.5	0.0	2.0	0.0	

- Where measure is the quantity that is monitored, such as:
 - VELOCITY - Vehicle longitudinal velocity
 - LAT_VELOCITY (VY) - Lateral velocity
 - VERT_VELOCITY (VZ) - Vertical velocity
 - LON_ACCEL - Vehicle longitudinal acceleration
 - LAT_ACCEL - Vehicle lateral acceleration
 - VERT_ACCEL (ACCZ) - Vertical acceleration

Specifying End Conditions

- DISTANCE - Total distance travelled by the vehicle during a mini-maneuver
- TIME - Simulation time
- YAW_ANGLE - Angular displacement about the vehicle's vertical axis
- YAW_RATE - Angular velocity about the vehicle's vertical axis
- YAW_ACCEL - Angular acceleration about the vehicle's vertical axis
- PITCH_ANGLE - Angular displacement about the vehicle's lateral axis
- ROLL_ANGLE - Angular displacement about the vehicle's longitudinal axis
- SIDE_SLIP_ANGLE - Angular offset between the projected vehicle's longitudinal axis and the vehicle's velocity vector
- LON_DIS (DX) - Longitudinal displacement
- LAT_DIS (DY) - Lateral displacement
- VERT_DIS (DZ) - Vertical displacement
- ROLL_RATE (WX) - Roll rate
- PITCH_RATE (WY) - Pitch rate
- ENGINE_SPEED - Angular velocity of the engine crankshaft in number of revolutions per minute (rpm)
- RACK_TRAVEL - Displacement in the steering rack joint
- STEERING_ANG - Angular displacement in the steering wheel joint

Specifying End Conditions

- **The current choices for measure are:**
 - test - Can be:
 - ‘==’ equal to (+/- 0.2%. For example, if the end condition for a mini-maneuver is VELOCITY and the value is 10 m/s, the Driving Machine terminates the mini-maneuver when the measured vehicle longitudinal velocity is greater than 9.98 m/s and less than 10.02 m/s.)
 - ‘>>’ greater than
 - ‘<<’ less than
 - value - The value against which the measure is tested to determine if the end condition is satisfied. Except for ENGINE_SPEED, you must specify the value in modeling units as defined in the UNITS block of the driver control file.
 - allowed_error - The allowed difference between the measure and value that still satisfies the test. allowed_error must be positive and be specified in modeling units as defined in the UNITS block of the driver control file (except ENGINE_SPEED).

Specifying End Conditions

- filter_time - The test must be satisfied continuously over the filter time to satisfy the end condition. filter_time must be positive.
- delay_time - Once the end condition is satisfied, delay the end of the mini-maneuver by delay_time.
- group - You specify a name to group end conditions together. All end conditions having the same group must be satisfied simultaneously to end a mini-maneuver. For example you might specify two end conditions:
 - Longitudinal velocity equal to 20 m/s
 - Lateral acceleration greater than 5 m/s/s
- Then you place the specified end conditions in the group 'mygroup'. To end the mini-maneuver, the longitudinal velocity must be 20 m/s and the lateral acceleration must be greater than 5 m/s/s. For example:

VELOCITY '==' 20 0.1 0.0 0.0 'mygroup'

LAT_ACCEL '>>' 5 0.1 0.0 0.0 'mygroup'

Extensible End and Abort Conditions

- A condition sensor is a user-defined element that consists of a data element array and strings. It references an existing variable class element (data element variable or measure solver computed), which is then tied to the label and unit strings by the array. The array also encapsulates a request (for plotting convenience) and a units conversion factor.
- In essence, a condition sensor represents a relationship between a measurable solver quantity (the variable class object) and a string label identifier that can be used in an event file (.dcf) to define an end condition for Adams/Car full-vehicle analyses.
- **Use of Condition Sensors in Adams/Car**
 - Before each vehicle analysis, Adams/Car browses the assembly for condition sensor elements and updates the data element `end_conditions_array` with the derived list.
 - At the beginning of the simulation, the Standard Driver Interface (SDI) uses the specified end condition measure string in the event to identify the associated variable class object in the dataset, which calculates the quantity the end condition sensor should compare to the target value.

Specifying Attributes

- To define the different sub-blocks, you must specify their attributes.
- The following table lists the attributes you must specify for each sub-block:

These sub-blocks:	Require these attributes:
Steering	<ul style="list-style-type: none">• ACTUATOR_TYPE• METHOD• MODE*
Throttle, brake, gear, and clutch	<ul style="list-style-type: none">• METHOD• MODE*

* Only required for the open-loop control method.

Specifying Attributes

- **Specifying an actuator type**

- When defining a STEERING sub-block, you must specify its actuator type.
- You use the actuator type to specify whether the Driving Machine steers the vehicle at the steering wheel or steering rack and whether the Driving Machine uses a force or motion, as described next.
- Actuators include rotation (wheel), translation (rack), force (rack), and torque (wheel).

- **Specifying a control method**

- When defining any sub-block, you must specify the control method that you want to use. You can choose from three control methods, as follows:
 - OPEN
 - MACHINE
 - SmartDriver

Specifying Attributes

- **OPEN_CONTROL**

- When you specify METHOD = 'OPEN' for steering or any other signal, the Driving Machine output for that signal is a function of time, and you must specify the function using the options in the CONTROL_TYPE argument.
 - **Arguments:**
 - CONTROL_TYPE = 'CONSTANT' || 'STEP' || 'RAMP' || 'IMPULSE' || 'SINE' || 'SWEPT_SINE' || 'DATA_DRIVEN' || 'FUNCTION'

- **MACHINE_CONTROL**

- The MACHINE_CONTROL option in the event file specifies the vehicle path, speed profile, and other parameters used by machine control for the mini-maneuver block in which it resides.

Specifying Attributes

- We recommend that if you use machine control for throttle, you also use machine control for braking.
- Also, if you use machine control for gear, you should use machine control for clutch.
- When you select **MACHINE_CONTROL** for gear and clutch, you must also supply the maximum and minimum engine speed. Machine control will up-shift to keep the engine speed less than the maximum, and down-shift to keep the engine speed greater than the minimum.
- If **SPEED_CONTROL = LAT_ACCEL**, then you must set **STEERING_CONTROL** to **SKIDPAD**.
- Arguments:
 - **STEERING_CONTROL = 'FILE' || 'STRAIGHT' || 'SKIDPAD'**
 - **SPEED_CONTROL = 'FILE' || 'LAT_ACCEL' || 'MAINTAIN' || 'VEL_POLYNOMIAL' || 'LON_ACCEL'**

Specifying Attributes

- **Specifying a mode**

- When defining any sub-block and you are using METHOD='OPEN', you must define MODE to be relative or absolute.
- With all open-loop maneuvers, the value of the preceding mini-maneuver is used as the starting point for the next mini-maneuver. This is irrespective of whether you choose relative or absolute. For example, your vehicle is driving on a skid pad at 30 mph, with 20° of steering wheel, when the first mini-maneuver is finished. The steering wheel angle for the next mini-maneuver will be 20°.
- The relative and absolute methods allow you to define what the steering wheel angle will be at the end of the next mini-maneuver.

Driver Control Data File Architecture

- Driver control data files may contain both OPEN-LOOP and CLOSED-LOOP blocks.
- The following table summarizes the closed-loop data that a .dcd file may contain:
 - The columns represent speed-control options from the driver parameters array.
 - The rows represent the steering control options from the driver parameters array.
 - The intersections give the data contained in the .dcd file and, therefore, the data input to the funnel to produce {x, y, lon_vel} as needed by Driving Machine.

Driver Control Data File Architecture

SPEED_CONTROL STEERING_CONTROL	none	lon_vel (p1=0)	lon_acc (p1=1)	lat_acc (p1=2)	path (p1=3)
none	NOT VALID	{{(distance or time), lon_vel}}	{{(distance or time), lon_acc}}	NOT VALID	NOT VALID
curvature (p1 = 0)	{distance, curvature}	{{(distance or time), curvature, lon_vel}}	{{(distance or time), curvature, lon_acc}}	{{(distance or time), curvature, lat_acc}}	NOT VALID
path (p1 = 1)	{x, y}	{x, y, lon_vel}	{x, y, lon_acc}	{x, y, lat_acc}	{x, y, time}
lat_acc (p1 = 2)	NOT VALID	{distance or time, lat_acc, lon_vel}	{distance or time, lat_acc, lon_acc}	NOT VALID	NOT VALID

APPENDIX D

WORKING WITH ROAD BUILDER

Road Builder

- In this workshop, you will learn the basics of creating a road data file using the Road Builder tool available in Adams/Car. Road data can be created with multiple segments, each segment representing predefined formulations like Linear, Curvature, and Transition Curve or through User Defined Functions and User Defined Points.
- The example contains the following sections:
 - Creating a Road data property file from scratch



This workshop takes one hour to complete.

Road Builder

- **Creating a Road data property file from scratch**
 - Below are the steps that you need to follow to generate a road data property file based on your own road profile. You need not have any assembly open when building a road profile.
- **Steps to create a road file from scratch using the Road Builder:**
 - Start **Adams/Car** in the **Standard Interface Mode**.
 - From the **Simulate** menu, point to **Full-Vehicle Analysis**, and then select **Road Builder**.
 - To start with, Adams displays the Road Builder with an example road data property file “**road_3d_sine_example.xml**” from the acar_shared database.
 - In the Road Builder dialog box, select the **Road Generator** tab.

Road Builder

- Add “segment1” to the table by entering **segment1** in the **Name** field and clicking the **Add** push button.

The screenshot shows the 'Road Builder' application window. The 'Road File' is set to 'file:///C:/Users/hbhardwaj/road_3d_sine_example.xml'. The 'Road Points' tab is active, displaying a table with the following data:

Name	Type	Number of Points	Start Point	Center Point	Tangent Point	End Point	Radius	Arc Length	Width
segment1	User Points	N/A	N/A	N/A	N/A	N/A	N/A	N/A	5.0

Below the table, the 'Add' button is circled in red. The 'Name' field contains 'segment1', and the 'Type' dropdown is set to 'RoadSegment'. The 'Parent' dropdown is set to 'Road3D'. The 'Export Points to Data Table' button is also visible.

Road Builder

- Double-click on “**segment1**” in the table to see the property editor as shown below.
- For segment1, **modify** the following
 - a. Type to Linear**
 - b. Start Point to -3300, 400, 0**
 - c. End Point to -10, 400, 0**

Note: you aren't setting 'width' now because you'll be setting it later on for several segments in the table view.

Road Builder

Road Builder

File Settings

Road File file:///C:/Users/hbhardwaj/road_3d_sine_example.xml

Header Global Soil Properties Road Points Obstacle Road Generator

↩ Name segment1 Comment

☒ Linear ☐ User Function

☐ Curvature ☐ Transition

☐ User Defined Points

Number of Points 50

Start Point -3300,400,0

End Point -10,400,0

Width 5.0

Bank 0.0

mu Left 1.0


mu Right 1.0

Export Points to Data Table

Current Field Unit length (meter)

Current Active File C:/Users/hbhardwaj/road_3d_sine_example.xml Save Save As Cancel

Road Builder

- Click on the  arrow to see the table view of the segment(s).
- Add new segments named: **segment2**, **segment3** and **segment4**.
- For **segment2**, modify the following columns (as shown in the following pages):
 - a. Type to Transition**
 - b. Segment 1 to segment1**
 - c. Segment 1 Point to End**
 - d. Segment 2 to segment3**
 - e. Segment 2 Point to Start**

Road Builder

Road Builder

File Settings

Road File

Header | Global | Soil Properties | Road Points | Obstacle | Road Generator

Name Filter

Name	Type	Number of Points	Start Point	Center Point	Tangent Point	End Point	Radius	Arc Leng
segment1	Linear	50	-3300.0, 400.0, 0.0	N/A	N/A	-10.0, 400.0, 0.0	N/A	N/A
segment2	User Points	N/A	N/A	N/A	N/A	N/A	N/A	N/A
segment3	Curvature	N/A	N/A	N/A	N/A	N/A	N/A	N/A
segment4	Linear	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Transition	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	User Function							
	User Points							

Add Name Type Parent

Export Points to Data Table

Current Field Unit

Current Active File Save Save As Cancel

Road Builder

Road Builder

File Settings

Road File

Header | Global | Soil Properties | Road Points | Obstacle | Road Generator

Name Filter

Name	Friction Left	Friction Right	Segment 1	Segment 1 Point	Segment 2	Segment 2 Point	User Points	Function
segment1	1.0	1.0	N/A	N/A	N/A	N/A	N/A	N/A
segment2	1.0	1.0	segment1	End	segment3	Start	N/A	N/A
segment3	1.0	1.0	N/A	N/A	N/A	N/A	0,0,0	N/A
segment4	1.0	1.0	N/A	N/A	N/A	N/A	0,0,0	N/A

Name
 Type
 Parent

Current Field Unit

Current Active File

Road Builder

- For **segment3**, modify the following (double-click to see view shown below):
 - Type to Linear**
 - Start Point to -10, -400, 0**
 - End Point to -3200, -400, 0**

The screenshot shows the 'Road Builder' application window. The 'Road File' is set to 'file:///C:/Users/hbhardwaj/road_3d_sine_example.xml'. The 'Header' tab is selected, and the 'segment3' is being edited. The 'Type' is set to 'Linear'. The 'Start Point' is '-10.0, -400.0, 0.0' and the 'End Point' is '-3200.0, -400.0, 0.0'. The 'Width' is '5.0', 'Bank' is '0.0', 'mu Left' is '1.0', and 'mu Right' is '1.0'. The 'Number of Points' is '50'. The 'Export Points to Data Table' button is visible. The 'Current Field Unit' is 'length (meter)' and the 'Current Active File' is 'C:/Users/hbhardwaj/road_3d_sine_example.xml'. The 'Save', 'Save As', and 'Cancel' buttons are at the bottom right.

Field	Value
Name	segment3
Type	Linear
Start Point	-10.0, -400.0, 0.0
End Point	-3200.0, -400.0, 0.0
Width	5.0
Bank	0.0
mu Left	1.0
mu Right	1.0
Number of Points	50

Road Builder

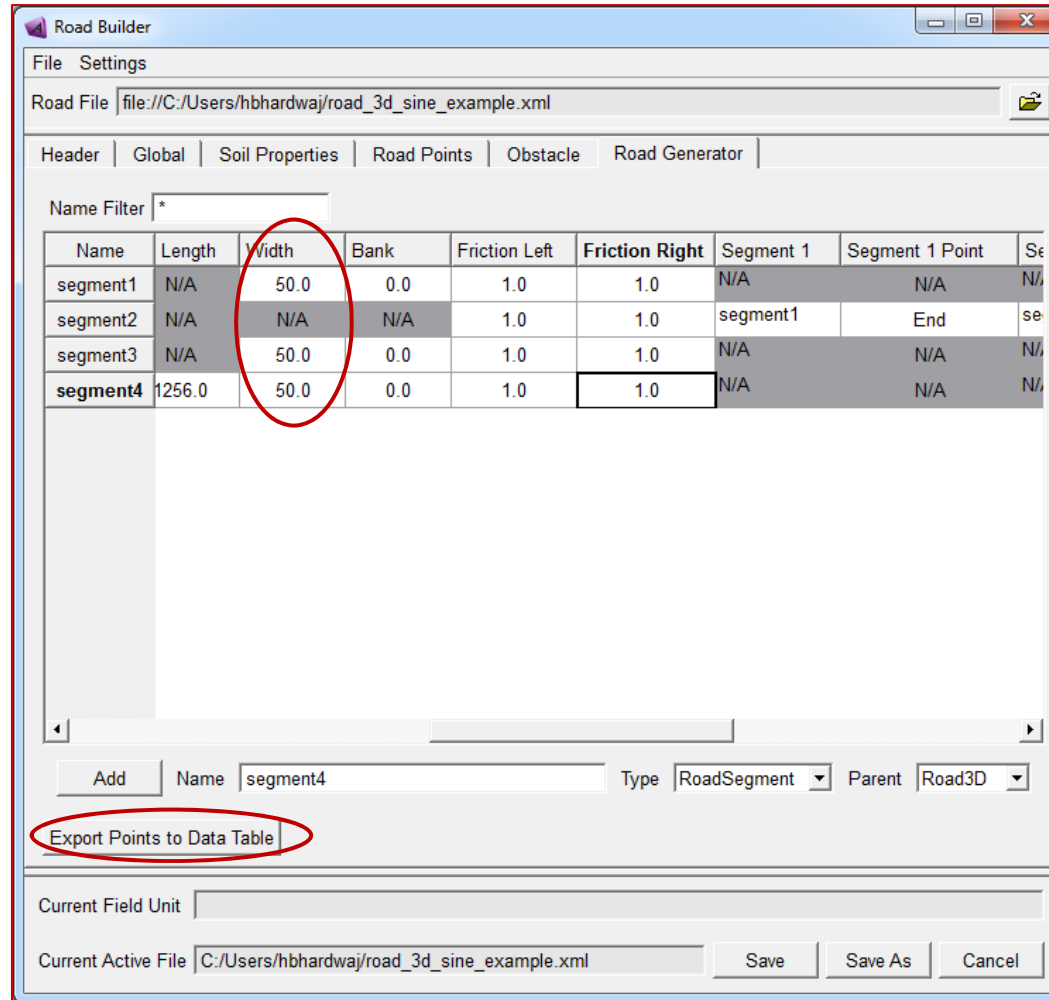
- For **segment4**, modify the following:
 - a. Type to Curvature**
 - b. Start Point to -3300, -400.0, 0.0**
 - c. Center Point to -3300, 0.0, 0.0**
 - d. Tangent Point to -3000, -400.0, 0.0**

The screenshot shows the 'Road Builder' application window. The 'Settings' tab is active, and the 'Road Points' sub-tab is selected. The 'Name' field is set to 'segment4'. The 'Type' section has four radio buttons: 'Linear', 'Curvature' (which is selected), 'User Defined Points', 'User Function', and 'Transition'. Below this, there are several input fields for defining the road segment: 'Number of Points' (50), 'Start Point' (-3300, -400.0), 'Center Point' (-3300, 0.0), 'Tangent Point' (-3000, -400.0), 'Radius' (400.0), 'Arc Length' (1256.0), 'Width' (5.0), 'Bank' (0.0), 'mu Left' (1.0), and 'mu Right' (1.0). At the bottom, there is an 'Export Points to Data Table' button, a 'Current Field Unit' dropdown set to 'length (meter)', and a 'Current Active File' field showing 'C:/Users/hbhardwaj/road_3d_sine_example.xml' with 'Save', 'Save As', and 'Cancel' buttons.

Road Builder

- Using the table view, you can see all of the segments at one time, set the **Width** to **50** for all segments.
- Click on the **Export Points to Data Table** push button to generate new road points and then select the **Road Points** tab to see the road points.

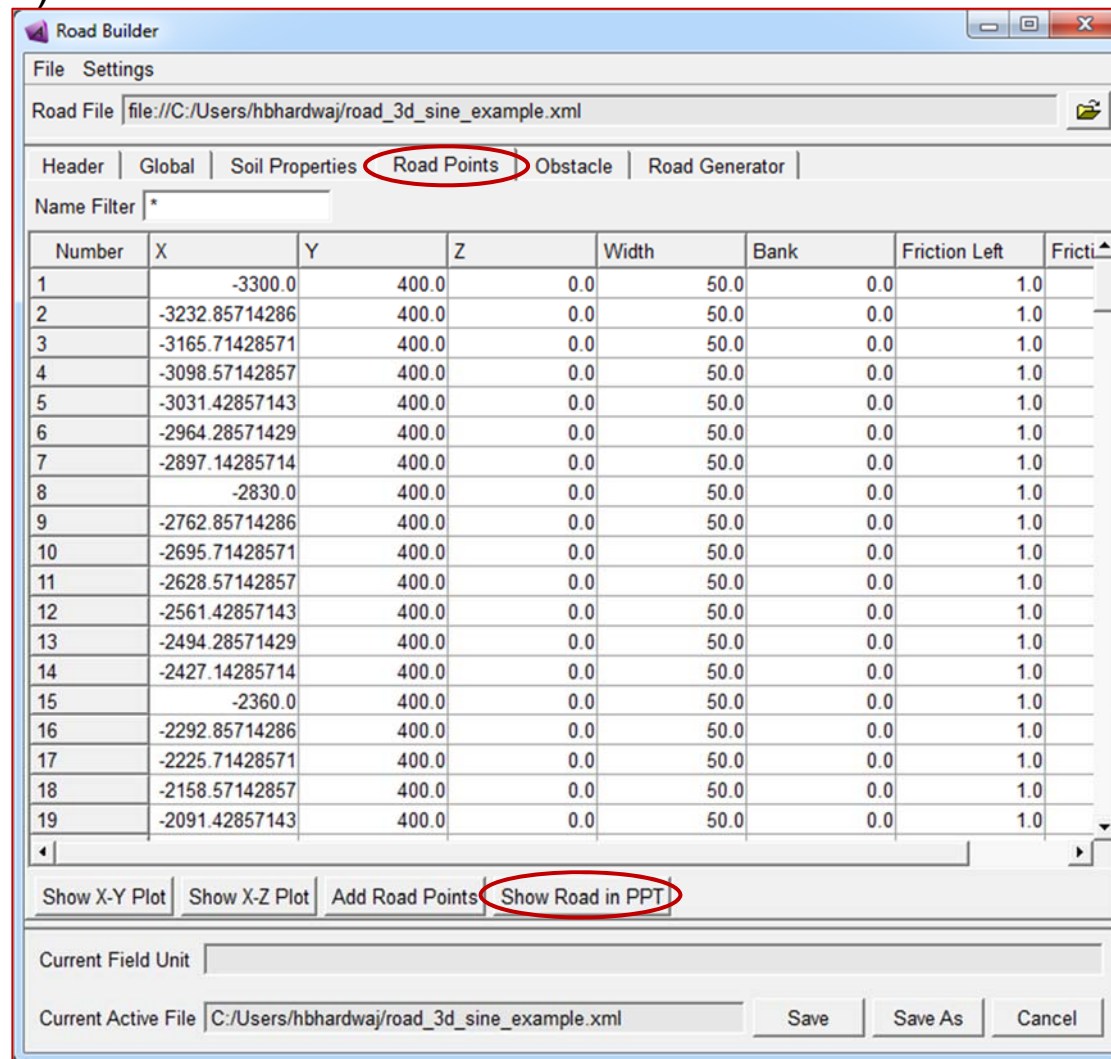
Road Builder



Tip: The **Export Points to Data Table** push button causes the segments information to be processed for generating road points.

Road Builder

- Click on the Show road in PPT push button (also available from the Road Points tab) to visualize the road in Adams/Car.



Road Builder

- Click on **Save As** to save the file as “road_file_example.xml”.



APPENDIX E

USING THE TIRE DATA FITTING TOOL (TDFT) IN ADAMS/CAR

Using the TDFT Tool

- In this workshop, you will learn how to use the Tire Data and Fitting Tool (TDFT) to calculate PAC2002 tire model parameters out of tire measurement data in Adams/Car.
- **The workshop contains the following sections:**
 - Perform PAC2002 tire parameter identification
 - Plotting tire characteristics



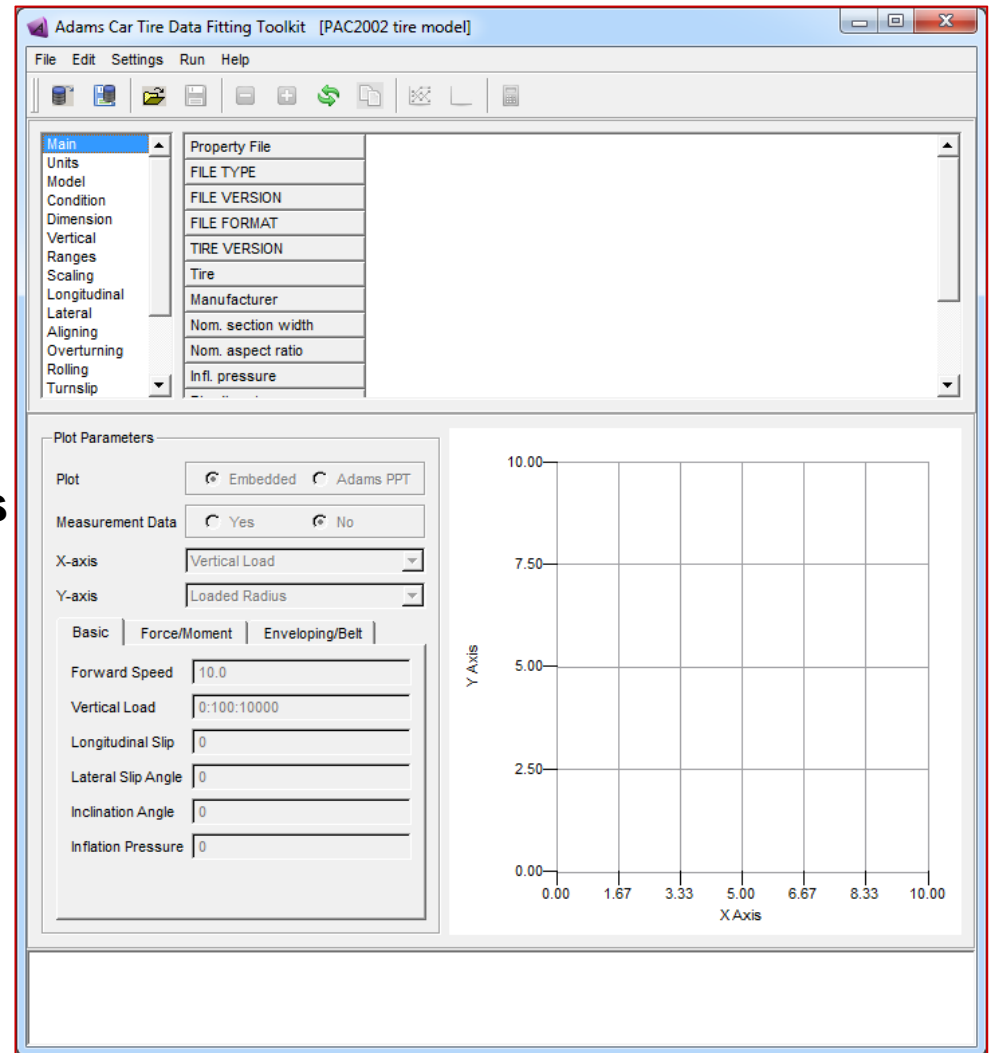
This workshop takes one hour to complete.

Perform PAC2002 tire parameter identification

- Perform PAC2002 tire parameter identification:
- The use of Tire Data Fitting Tool (TDFT) to calculate and visualize PAC2002 tire model parameters out of tire measurement data or virtual data for steady state pure and combined slip conditions has been demonstrated using tire property file “pac2002_205_55R16_tdft.tir”.
- You can also use an empty tire property file that does not have tire parameter identification or fitting data, so by default template file (tdft_template.tir) will be loaded to define tire parameter identification data. An example measurement data file (Fm_data_example_tdft.txt) containing measurement data for both pure and combined slip conditions has been used from “shared_acar_database.cdb” database.

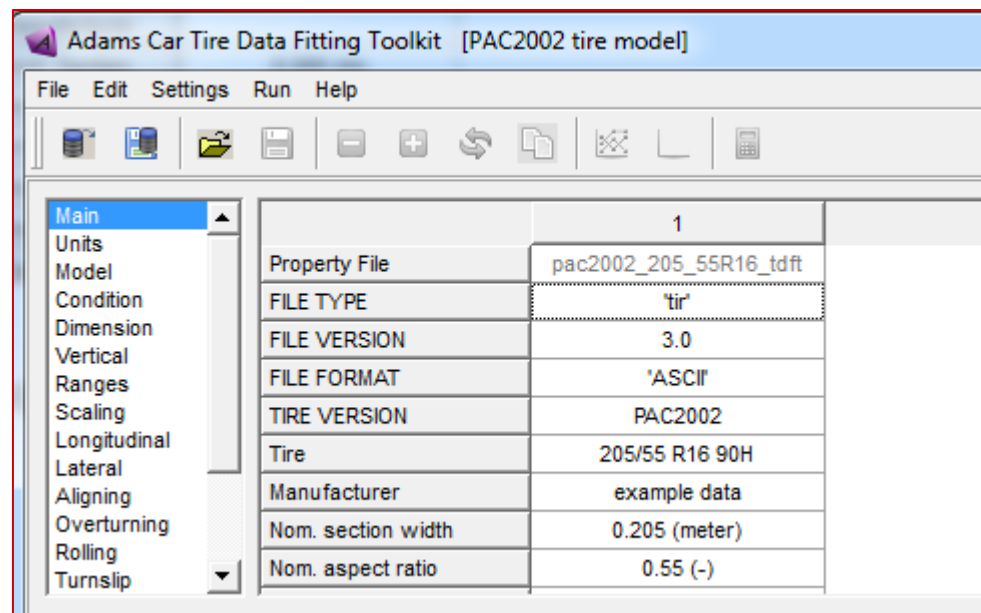
To start the Tire Data and Fitting Tool in Adams/Car:

- To start the Tire Data and Fitting Tool:
 1. Open Adams/Car
 2. Click on **Standard Interface mode**
 3. Select **Simulate** menu
 4. Point to **Component Analysis** and select **Tire Data and Fitting Tool**



Opening the tire property file

- Opening the Tire property file and adding to the database:
 - Select **File** menu
 - Click on **Open Tire Property and Add to Database**
 - Open the example tire property file **pac2002_205_55R16_tdft.tir** located in **<adams_install>/acar/shared_car_database.cdb/tires.tbl**



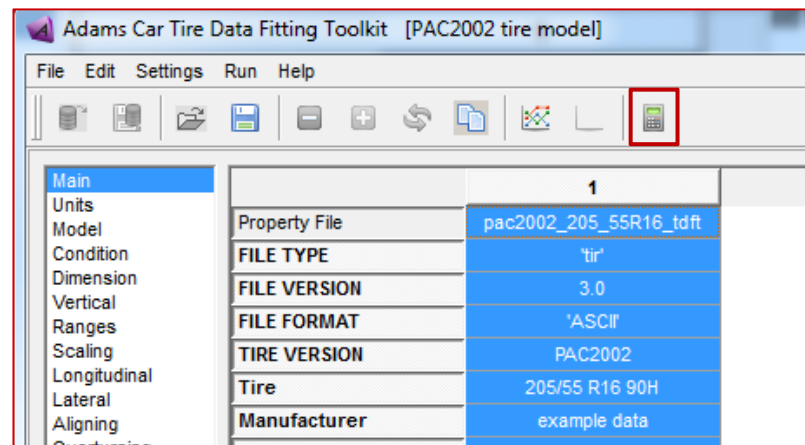
Note: The warning message about updating header column can be ignored.

Starting the Tire Fitting Tool

- **To start the Tire Fitting Tool:**
 - Verify that following parameters have been specified:
 - Under Tab Dimension/Vertical:
 - **FNOMIN** (\$Nominal wheel) and
 - **UNLOADED RADIUS** (\$Free tire radius)

Note: The above parameters are used in the PAC2002 tire model to make the tire parameters dimensionless.

- Select the tire by clicking the column in **Main GUI**.
- Click on the Tire Fitting icon to start the **Tire Fitting Tool**.
- Message pops up indicating that no tire parameter identification data is present yet. Press **OK**.

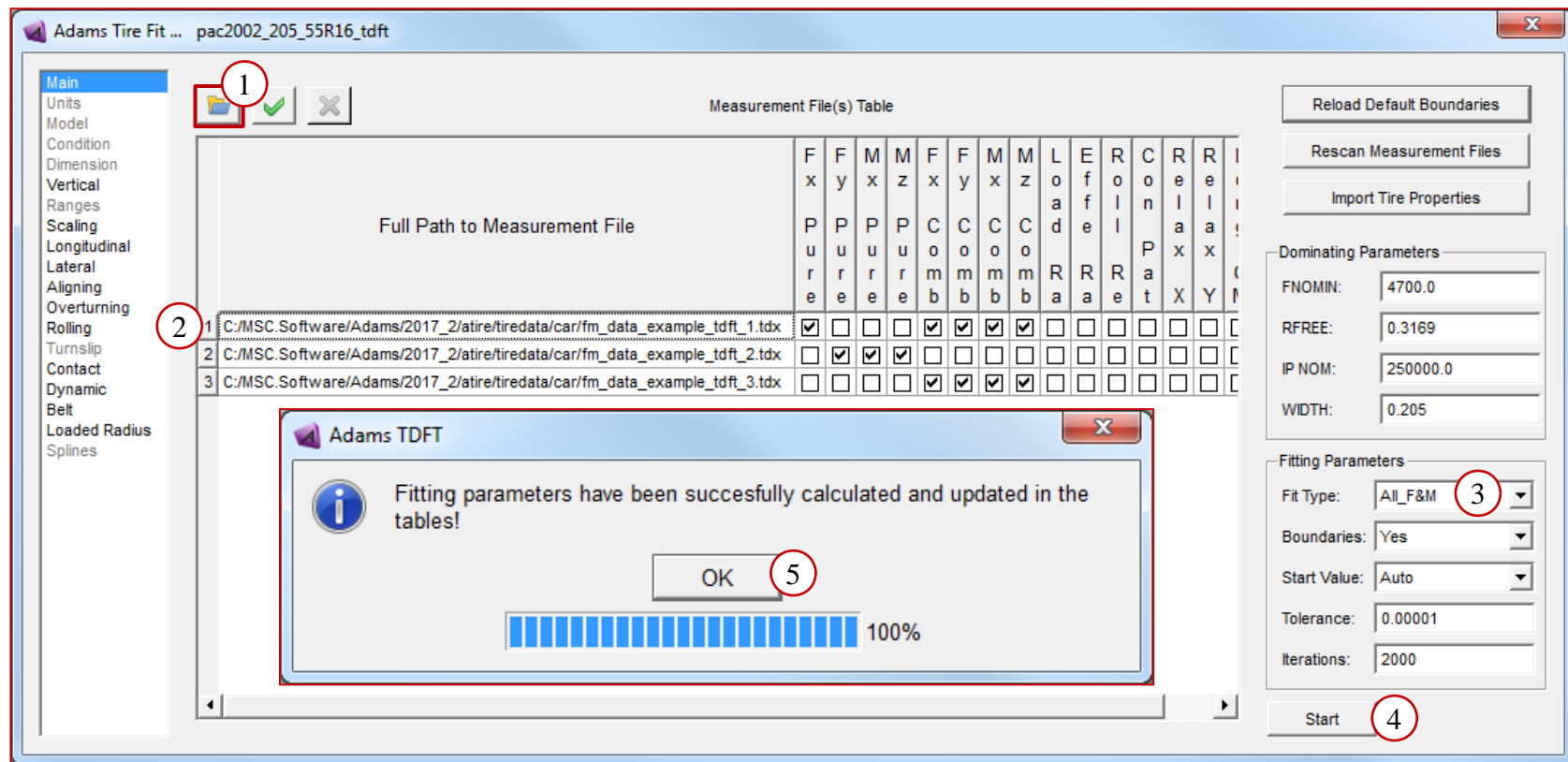


Adding Measurements


- Once the step 4 is completed. TDFT toolkit will launch another dialog box in order to enter the measurement data.
- **Adding Measurement data:**
 1. Click on **Add Measurements by file or directory tab**.
 2. Browse location **C:\MSC.Software\Adams\2017_2\tire\tiredata\car** and select **fm_data_example_tdft_1.txt fm_data_example_tdft_2.txt fm_data_example_tdft_3.txt** (example measurement data file containing steady-state force and moment measurement data for pure and combined slip conditions).
 3. Set the **Fit Type** under **Fitting Parameters** as **All_F&M**.
 4. Press **Start** to run the parameter identification.

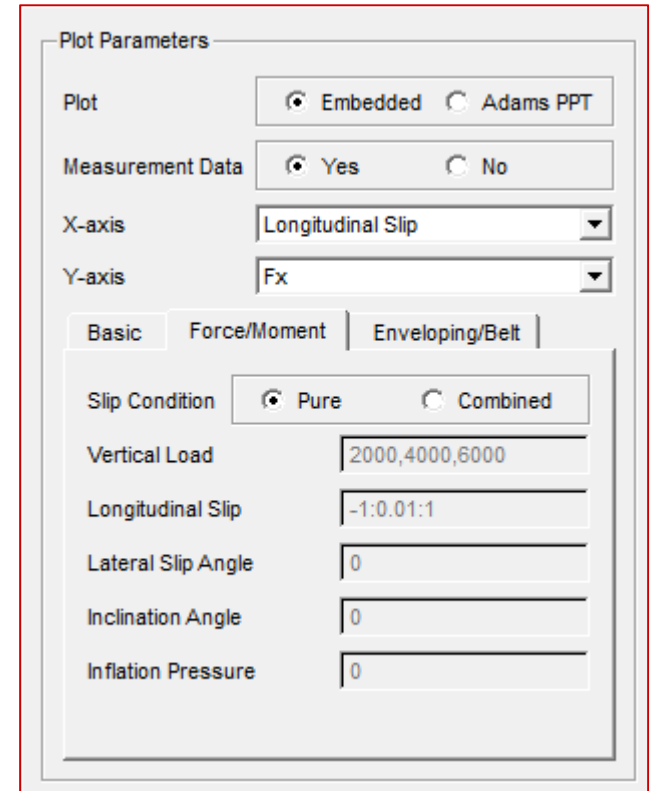
Adding Measurements

5. After completion of the fitting parameters calculation, Tire toolkit will show following message. Click **OK**.



Plotting the results

- To plot the results cancel the last tire fit window and return to the Tire Toolkit window. Plot the results as follows:
 1. Go to **Plot Parameters**
 2. Select **embedded** plot to display the plot within the tool (or select Adams/PPT).
 3. Set **Measurement Data** to **Yes**
 4. Set **Slip Condition** to **Pure**
 5. Select the Tire Plot  tab to display the plot.



Plot Parameters

Plot ☒ Embedded ☐ Adams PPT

Measurement Data ☒ Yes ☐ No

X-axis Longitudinal Slip

Y-axis Fx

Basic Force/Moment Enveloping/Belt

Slip Condition ☒ Pure ☐ Combined

Vertical Load 2000,4000,6000

Longitudinal Slip -1:0.01:1

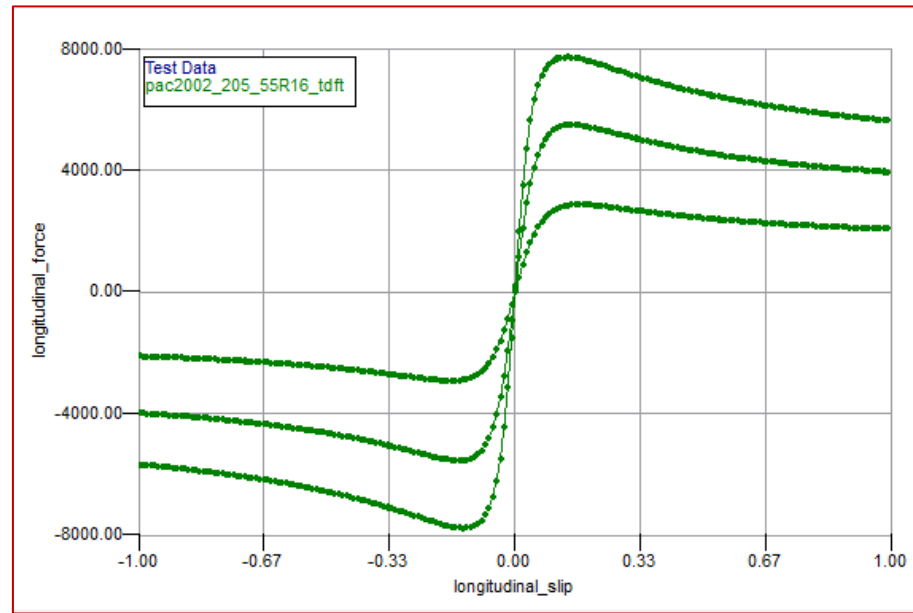
Lateral Slip Angle 0

Inclination Angle 0

Inflation Pressure 0

Plots (Embedded and PPT)

- The measurement data (added to the Tire Fitting Tool) is plotted against PAC2002 model data.
- We have successfully verified the resultant tire model parameters by comparing the measurement (test) data to the PAC2002 model data.
 - Note: **Measurement = No** means that measurement data (added in the Tire Fitting Tool) is not used for plotting but only input data entered in the Plot Tire Characteristic Tool is used to evaluate the PAC2002 tire model.





APPENDIX F

FULL-VEHICLE ASSEMBLY

Full-Vehicle Assembly

- **This workshop is intended to combine the knowledge you have attained throughout the course to create a full-vehicle assembly. The major steps for this workshop are described, but the exact steps are omitted. If you have questions, please refer to previous sections or ask the instructor.**
- **At this point, you should have the following templates done:**
 - MacPherson suspension (be sure to use the one you created rather than the one in the shared database, because these have topological differences)
 - Rack and pinion steering
 - You should also have access to `_double_wishbone.tpl` and `_handling_tire.tpl`, which is located in the shared database.
- **Before you build the assembly, you must create the body. You then attach the front and rear suspensions, together with the steering system, to the body. You also create appropriate communicators.**



This workshop takes one hour to complete.

Full-Vehicle Assembly

- **Creating the Body**

- In Template Builder, open the template `body_training.tpl`, which is in the `acar_training_MD.cdb` database you added to the session. This template has one rigid body (`ges_chassis`) that will act as the chassis.
- Creating and testing communicators

- Body versus MacPherson suspension (used as front suspension)

- Open the MacPherson template you created before.
 - Use Communicator Info to confirm that the front suspension hooks up to the body. For the MacPherson suspension, you have the following input communicators

<code>ci[lr]_strut_to_body</code>	<code>mount</code>
<code>ci[lr]_subframe_to_body</code>	<code>mount</code>
<code>ci[lr]_tierod_to_steering</code>	<code>mount</code>

- For the body, the communicators of interest are `strut_to_body` and `subframe_to_body`, which are mount communicators. You must convey to the front suspension what part the upper strut should attach to. In this case it's the body, since that is the only part in the chassis template.
 - Create a new output communicator in `_body_training.tpl` of type mount (name it `body`, assign to it either right or left symmetry), that will match the `ci[lr]_strut_to_body`, by entering the same matching name.
 - Edit the communicator, `body`, to have a matching name for `subframe_to_body`.
 - Test the communicators between the body and the MacPherson templates. Remember that you must have the templates open to be able to test them. Use a minor role of any for the body and front for the suspension.

Full-Vehicle Assembly

- **Body versus steering**

- Open the steering template. Here you will set up the steering and body template so that they will attach to each other. That is, make sure that the steering-column housing and rack housing are attached to the body. The rack housing should really be attached to a suspension subframe, but since you are not including a subframe, use the body instead.
- Display the body template and create an output communicator of type mount, single symmetry, with the name `body_single`, with a matching name of the corresponding input communicator, named `steering_column_to_body`, in the steering template.
- Modify the same single output communicator to include the matching name `rackhousing_to_body`.
- Test the communicators between the body and the steering. Use a minor role of any for the body, and front for the steering.

Full-Vehicle Assembly

- **Body versus double-wishbone suspension (used as rear suspension)**

- In this section, you will set up communicators so that the rear suspension hooks up to the body. For information about this template, look in the online help documentation.
 1. Open the double-wishbone suspension from the Adams/Car shared database.
 2. Select Build - Communicator - Info.
 3. Set Type to Input, set Entity to Many, and then select mount. Select OK.
- Adams/Car displays a list of all input communicators in the double-wishbone suspension template:
- Listing of input communicators in '_double_wishbone'

Communicator Name: Class: From Minor Role: Matching Name:

ci[lr]_strut_to_body	mount	inherit	strut_to_body
ci[lr]_tierod_to_steering	mount	inherit	tierod_to_steering
ci[lr]_tripot_to_differential	mount	inherit	tripot_to_differential
ci[lr]_uca_to_body	mount	inherit	uca_to_body
cis_subframe_to_body	mount	inherit	subframe_to_body

'_double_wishbone' contains:

9 input communicators of type 'mount'

Full-Vehicle Assembly

4. Perform a communicator test between the body (any) and double-wishbone suspension (rear), and confirm that the strut attaches to the body. This is the same communicator in the body template you created for the front suspension, so you don't have to create it again. It should pass the information to both the front and the rear mount parts.
5. Modify the left/right body communicator, named body, in the body to pass information to the uca_to_body input communicator located in the double-wishbone suspension template.
6. To keep the wheels going straight, hook up the tie rods to the body, so the wheels won't turn. Create a left/right mount communicator named body_rear. This time, instead of inherit for a minor role, choose rear, and enter tierod_to_steering as a matching name.
7. Modify the single body output communicator, named body_single, to include the matching name subframe_to_body for the double-wishbone suspension. The previous communicators are pretty straightforward, but let's investigate input communicators ci[lr]_tripot_to_differential in the double-wishbone suspension.
8. From the Tools menu, select Database Navigator.
9. At the top of the Database Navigator, select Associativity.
10. In the Filter text box, enter *tripot*, and then press Enter.

Full-Vehicle Assembly

11. Double-click `_double_wishbone` to see the modeling elements that contain the string `tripot` in their names.
 - You should see, among others:
 - `cil_tripot_to_differential`
 - `cir_tripot_to_differential`
12. Highlight either of these, and then, in the right window of the Database Navigator, at the bottom, select `Uses`. You should see the following:
 - `._double_wishbone.cir_tripot_to_differential.cir_tripot_to_differential =`
 - `._double_wishbone.ground.mtr_mount_j_5 (Marker)`
 - `._double_wishbone.cir_tripot_to_differential.cir_tripot_to_differential =`
 - `._double_wishbone.mtr_tripot_to_differential (Part)`
 - `._double_wishbone.cir_tripot_to_differential.cir_tripot_to_differential =`
 - `._double_wishbone.mtr_fixed_5 (Fixed Joint)`
 - `._double_wishbone.cir_tripot_to_differential.minor_role.minor_role =`
 - `._double_wishbone.mtr_tripot_to_differential.minor_role (Variable)`
13. In the Database Navigator, change Associativity to Topology by Parts.

Full-Vehicle Assembly

14. Select **mtl_tripot_to_differential**, which should show you the following:

Part mtl-tripot_to differential is connected to:

ground	via	mtl_fixed_5 (Fixed Joint)
gel_tripot	via	joltra_tripot_to_differential (Translational Joint)

Therefore, the tripot_to_differential mount part is connected to ge[lr]_tripot.

15. Select **gel_tripot**

Part gel-tripot is connected to:

mtl_tripot_to_differential	via	joltra_tripot_to_differential (Translational Joint)
gel_drive_shaft	via	jolcon_drive_sft_int_jt (Convel Joint)

16. Enter “*” for the filter and select **gel_drive_shaft**.

Part gel_drive_shaft is connected to:

gel_tripot	via	jolcon_drive_sft_int_jt (Convel Joint)
gel_spindle	via	jolcon_drive_sft_otr (Convel Joint)

17. You can use the Database Navigator again to look at the Associativity of the spindle (toggling **Is Used By**). Note that the spindle is used by an output communicator:

```
._double_wishbone.col_suspension_mount.col_suspension_mount =  
._double_wishbone.gel_spindle (Part)
```

This is the mount that should connect to the wheel. So, overall, you can see that the tripot_to_differential mount part should normally connect to a powertrain, if you wanted to use one. Because you are not including a powertrain for this exercise, we will ignore this mount part and let it connect to ground. In the Standard Interface, you will then deactivate the driveline so that the vehicle is not locked to ground, too.

Full-Vehicle Assembly

- **Double –wishbone suspension versus wheels**

- The double-wishbone suspension must pass the information on the part to which the wheels should attach. You must edit the mount output communicators in the doublewishbone suspension named `suspension_mount` to include the matching name `wheel_to_susp`.
- Body versus SDI_testrig
 1. To check the test rig, SDI_testrig, select Build -> Communicator -> Info. There is one mount input communicator that expects information from the body, `cis_body_subsystem`.
 2. Modify `cos_body_single` in the body template to include the `body_subsystem` matching name.
 3. Test all the communicators for all the templates that you will use for the full vehicle.
 - NOTE: When testing these communicators, you must specify what the templates' minor roles will be when you will build the subsystems. Assign minor roles to templates, as shown next:

Full-Vehicle Assembly

Template:

.__MDI_SDI_TESTRIG

._Handling_Tire

._Handling_Tire

._body_training

._double_wishbone

._steer_final

._macpherson

Minor role:

any

front

rear

any

rear

front

front

The following is communicator information for the mount and location classes in the templates *after* modifications in this workshop (one example; other variations will work, too, depending on implementation):

Listing of input communicators in '_handling_tire'

ci[lr]_suspension_mount	mount	inherit	suspension_mount
ci[lr]_suspension_upright	mount	inherit	suspension_upright
ci[lr]_wheel_center	location	inherit	wheel_center

Full-Vehicle Assembly

Listing of output communicators in '_handling_tire'

co[lr]_rotor_to_wheel	mount	inherit	rotor_to_wheel
-----------------------	-------	---------	----------------

Listing of input communicators in '_macpherson_final'

ci[lr]_strut_to_body	mount	inherit	strut_to_body
ci[lr]_subframe_to_body	mount	inherit	subframe_to_body
ci[lr]_tierod_to_steering	mount	inherit	tierod_to_steering

Listing of output communicators in '_macpherson_final'

co[lr]_wheel_center	location	inherit	wheel_center
co[lr]_suspension_mount	mount	inherit	suspension_mount, wheel_to_susp
co[lr]_suspension_upright	mount	inherit	suspension_upright

Listing of input communicators in '_double_wishbone'

ci[lr]_arb_pickup	location	inherit	arb_pickup
ci[lr]_strut_to_body	mount	inherit	strut_to_body
ci[lr]_tierod_to_steering	mount	inherit	tierod_to_steering

Full-Vehicle Assembly

ci[lr]_tripot_to_differential	mount	inherit	tripot_to_differential
ci[lr]_uca_to_body	mount	inherit	uca_to_body
cis_subframe_to_body	mount	inherit	subframe_to_body

Listing of output communicators in '_double_wishbone'

co[lr]_tripot_to_differential	location	inherit	tripot_to_differential
co[lr]_wheel_center	location	inherit	wheel_center
co[lr]_arb_bushing_mount	mount	inherit	arb_bushing_mount
co[lr]_droplink_to_suspension	mount	inherit	droplink_to_suspension
co[lr]_suspension_mount	mount	inherit	suspension_mount, wheel_to_susp
co[lr]_suspension_upright	mount	inherit	suspension_upright
cos_engine_to_subframe	mount	inherit	engine_to_subframe
cos_rack_housing_to_suspension_subframe	mount	inherit	rack_housing_to_suspension_subframe

Listing of input communicators in '_steer_final'

cis_rackhousing_to_body	mount	inherit	rackhousing_to_body
-------------------------	-------	---------	---------------------

Full-Vehicle Assembly

cis_steering_column_to_body	mount	inherit	steering_column_to_body
ci[lr]_tierod_to_steering	mount	inherit	tierod_to_steering
ci[lr]_tripot_to_differential	mount	inherit	tripot_to_differential
ci[lr]_uca_to_body	mount	inherit	uca_to_body
cis_subframe_to_body	mount	inherit	subframe_to_body

Listing of output communicators in '_steer_final'

co[lr]_tierod_to_steering	mount	inherit	tierod_to_steering
---------------------------	-------	---------	--------------------

Listing of input communicators in '_body_final'

0 input communicators of type 'location'

0 input communicators of type 'mount'

Listing of output communicators in '_body_final'

co[lr]_body	mount	inherit	strut_to_body, subframe_to_body, uca_to_body
-------------	-------	---------	--

Full-Vehicle Assembly

co[lr]_body_rear	mount	rear	tierod_to_steering
cos_body_single	mount	inherit	steering_column_to_body, rackhousing_to_body, subframe_to_body, body_subsystem

Listing of input communicators in '_MDI_SDI_TESTRIG'

ci[lr]_front_suspension_mount	mount	front	suspension_mount
ci[lr]_rear_suspension_mount	mount	rear	suspension_mount
cis_body_subsystem	mount	inherit	body_subsystem
ci[lr]_wheel_center_front	location	front	wheel_center
ci[lr]_wheel_center_rear	location	rear	wheel_center

Listing of output communicators in '_MDI_SDI_TESTRIG'

cos_std_tire_ref	location	any	std_tire_ref
------------------	----------	-----	--------------

- Save all your templates
- Create new subsystems for all your templates.

Full-Vehicle Assembly

- Create a new full vehicle assembly with the subsystems you just created. Confirm that none of the critical communicators appear in the warning message about unmatched communicators. Otherwise, close the assembly and return to Template Builder to make the necessary changes. Adams/Car displays the following message about unassigned communicators:

testrig.cis_engine_rpm
testrig.cis_engine_speed
testrig.cis_max_gears
testrig.cis_diff_ratio
testrig.cis_max_throttle
testrig.cis_max_brake_value
testrig.cis_engine_speed_limit
testrig.cis_engine_stall_speed
testrig.cis_transmission_spline
testrig.cis_clutch_displacement_ic
testrig.cis_max_engine_driving_torque
testrig.cis_max_engine_braking_torque
ci[lr]_ARB_pickup
ci[lr]_tripot_to_differential (attached to ground) ...

Note that **ci[l,r]_tripot_to_differential** has attached to ground. You will deactivate the driveline so that it is not used, since you have not included a powertrain, and the vehicle will not be attached to ground.

Full-Vehicle Assembly

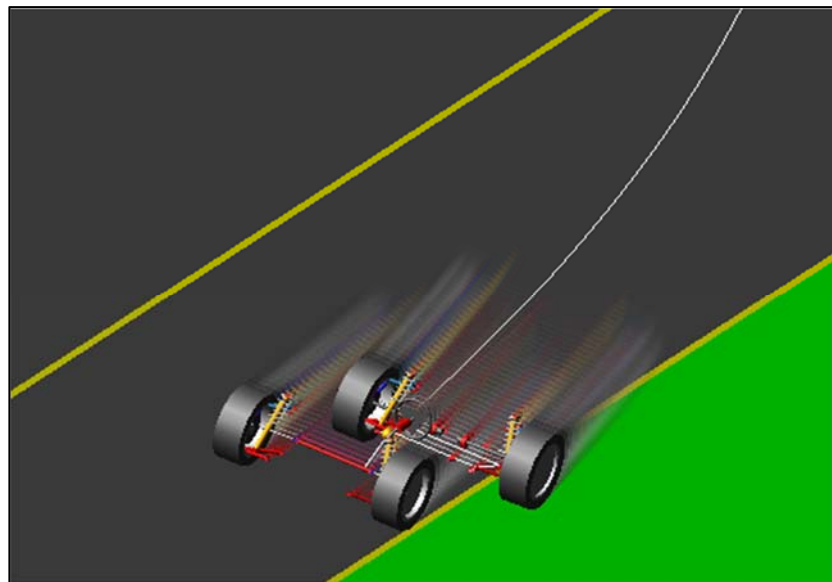
- Adjust the different subsystems so they appear in the right location relative to each other. Shift the front and the rear suspensions by this amount:
 - Front suspension aft 250 mm
 up 230 mm
 - Rear suspension aft 2800 mm
 up 37.30 mm
 - Steering aft 250 mm
 up 230 mm
- From the **View** menu, select **Subsystem**, and then select the double-wishbone suspension in your assembly.
- From the **Adjust** menu, select **Driveline Activity**, and then set Current Mode to Inactive. This turns off the driveline in the rear suspension, which removes the connection of the vehicle to ground.
- **Select View -> Assembly.**
- **Define a preload:**
 - Right-click either of the front springs, and then select **Modify**.
 - Set **Installed Length** to **135**.
 - Select **OK**.
- **Change the front dampers:**
 - Right-click either of the front dampers, and then select **Modify**.
 - Set the damper to **<acar_shared>/dampers.tbl/mdi_shk_0001.dpr**.
 - Select **OK**.

Full-Vehicle Assembly

- **Run any full-vehicle open-loop steering analysis. In the open-loop steering analyses, because the assembly has no powertrain, there will be no power to the wheels. You can, however, run these events with initial velocity.**

APPENDIX G

ADAMS/CAR MECHATRONICS



Adams/Car Mechatronics

- You can use Adams/Car Mechatronics to easily incorporate control systems into your vehicle models. It contains modeling elements which transfer information to/from the control system.
- The benefit of using Adams/Car Mechatronics is that it shares data and simulation components with Adams/Car, as well as the control system development code you are using. It means that complete system-level optimization is made easier for complex problems such as shunt-and-shuffle, ABS impact on ride/judder and so on.
- You can load the Adams/Car assembly containing a control system as any other assembly, without having to perform several manual tasks to run the coupled simulation.
- Adams/Car Mechatronics includes a Signal Manager for setting up the connection between the transducers/actuators and the control systems.

Benefits of Adams/Car Mechatronics

- **Using Adams/Car Mechatronics you can:**
 - **Explore the performance** of your coupled mechanical/controls design and refine your coupled design before building and testing a physical prototype.
 - **Analyze design changes** much faster and at a lower cost than physical prototype testing would require. For example, you can change springs with a few mouse clicks instead of waiting for a mechanic to install new ones in your physical prototype before re-evaluating your design.
 - **Vary the kinds of analyses** faster and more easily than if you had to modify instrumentation, test fixtures, and test procedures.
 - Work in a more secure environment without the fear of losing data from instrument failure or losing testing time because of poor weather conditions.
 - Run analyses and what-if scenarios without the dangers associated with physical testing.

Methods of linking Control System

- **Adams/Car Mechatronics provides three different methods to link the control system to the vehicle model.**
 - Controls System based on Function Expressions.
 - Compiled Control System Representation – Use Controls code generated by software such as Easy5 or Matlab/Simulink.
 - Co-simulation – Adams/Solver solves the mechanical system equations and the control application (eg. Easy5 or Matlab) solves the control system equations.

Mechatronics Workshop

- **Problem Statement:**

- Use Adams/Car Mechatronics to investigate vehicle dynamics through a lane change event with and without an Electronic Stability Program (ESP) system active.

- **The Control systems will be based on either:**

- Adams/Solver function expressions
- Compiled control system libraries created in MATLAB/Simulink.



This workshop takes about two hours to complete.

Baseline Simulation

- **Import the model**
 - Start **Adams/Car** in Standard Interface mode.
 - Load the Adams/Car Mechatronics plugin using the menu picks: **Tools** → **Plugin Manager**.
 - Load the assembly named **MDI_Demo_Vehicle_It_ABS_ESP_01.asy** from the **amech_shared** database, which is automatically added to the session when the Mechatronics plugin is selected.

Baseline Simulation

- **Running a Baseline Simulation:**
- You will run a lane change simulation on a road surface having two very different traction characteristics: the left lane is slippery ($\mu = 0.2$) while the right lane has greater traction ($\mu = 0.9$). The ABS and ESP systems will be deactivated for the baseline simulation as the steering wheel is guided through a prescribed motion.
- **Deactivate the control systems:**
 - From the menus, select **Mechatronics** → **Control System** → **Modify...**
 - In the **Modify Control System** panel, right-click the **Name** dialog box and browse for the system named:
.MDI_Demo_Vehicle_It_ABS_ESP_01.ABS_system_01.ues_ABS
 - Set the **Active** to be off.
 - In a similar fashion, turn off the ESP system named:
.MDI_Demo_Vehicle_It_ABS_ESP_01.ESP_system_01.ues_ESP

Baseline Simulation

- Run a lane change simulation

1. From the menus, select **Simulate** → **Full-Vehicle Analysis** → **Open-Loop Steering Events** → **Single Lane Change**.
2. Specify the simulation parameters as following, taking care to use the Road **3d_road_smooth_flat_splitmu02.xml** Data File, provided in the **acar_training_MD/roads.tbl**.
3. Run the simulation.

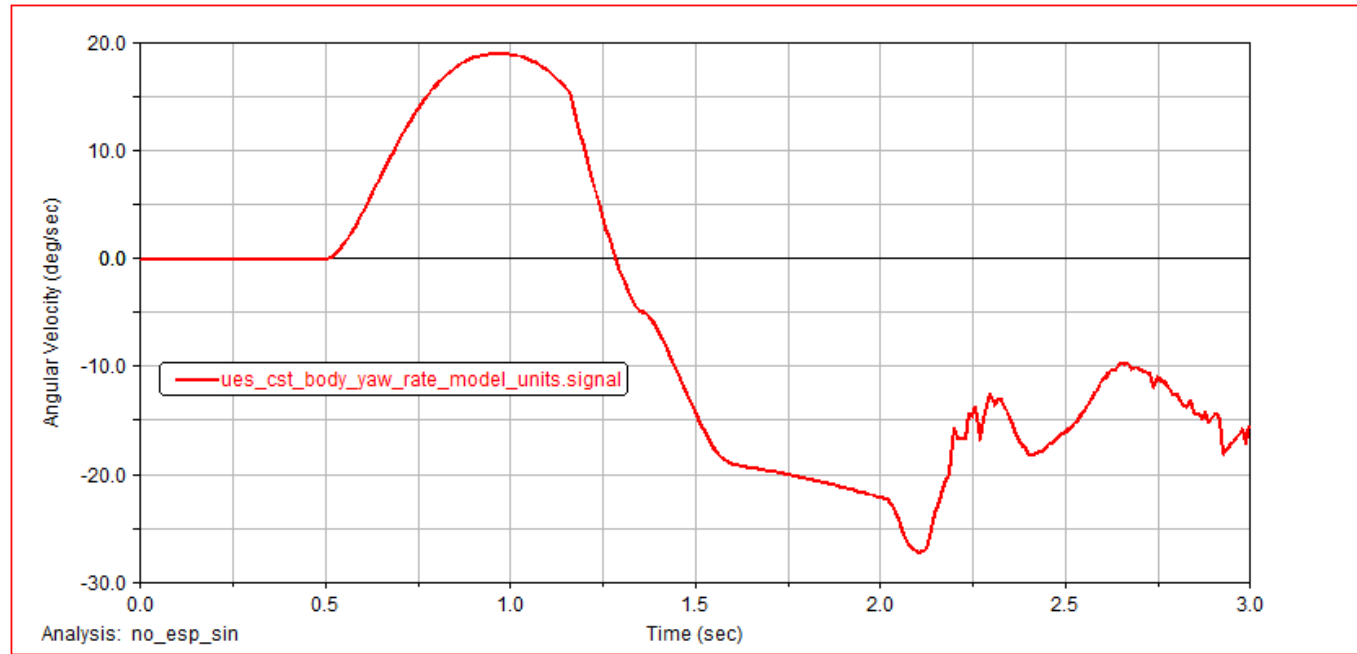
Full-Vehicle Analysis: Single Lane Change

Full-Vehicle Assembly	MDI_Demo_Vehicle_It_ABS_ESP_01
Assembly Variant	default
Output Prefix	no_esp
End Time	3
Number Of Steps	300
Simulation Mode	interactive
Road Data File	D:\acar_training_MD.cdb\roads.tbl\3d_road_smooth_flat_splitmu02.xml
Initial Velocity	75 km/hr
Gear Position	5
Maximum Steer Value	55
Start Time	0.5
Cycle Length	1.5
Steering Input	Angle
<input type="checkbox"/> Cruise Control	
<input checked="" type="checkbox"/> Quasi-Static Straight-Line Setup	
<input checked="" type="checkbox"/> Create Analysis Log File	

OK Apply Cancel

Baseline Simulation

4. Plot the vehicle yaw rate (`ues_cst_body_yaw_rate_model_units`), as follows



5. Note the increasing yaw rate after 1.5 seconds as the vehicle begins to spin out of control near the end of the event. (**Note:** that the steering wheel has a prescribed motion throughout the simulation, so no driver correction occurs to counteract the yawing).

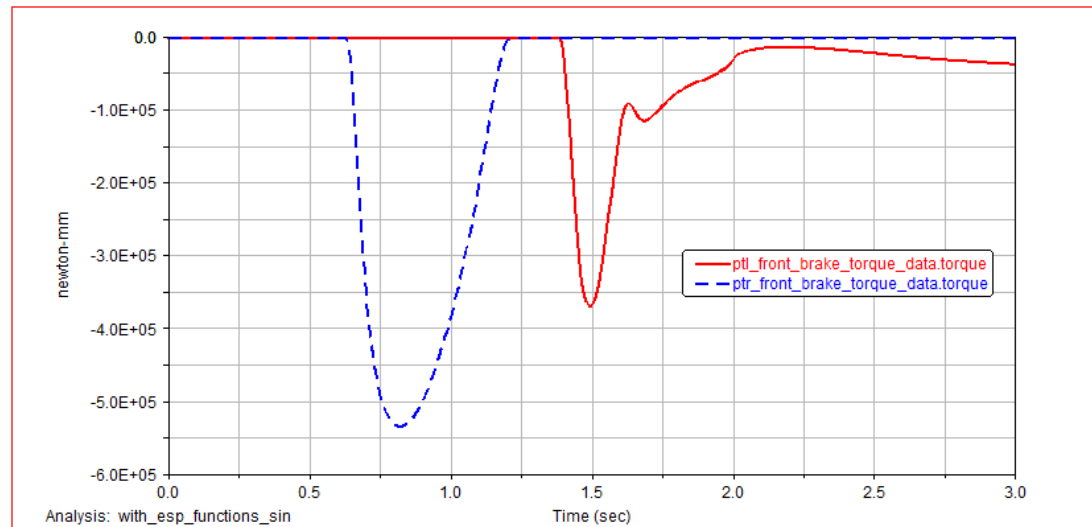
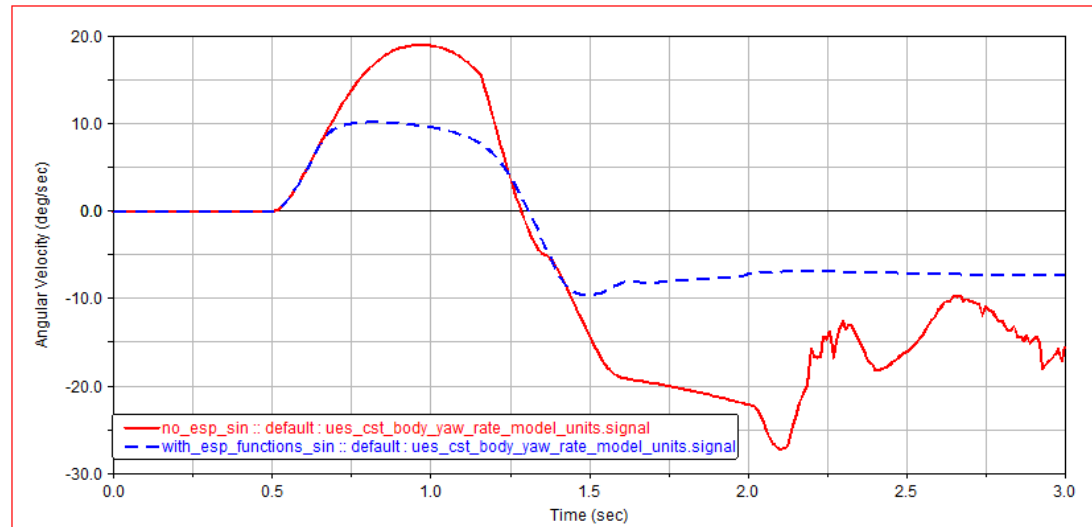
Function Expression Control System

- **Activating the 'Function Expression' ESP Controller**
 - Now activate the ESP and ABS controllers that are built into the Adams/Car Mechatronics demo model. These controllers are based on Adams/Solver Function Expressions alone, so neither MATLAB nor EASY5 are required.
- **Activate the control systems**
 1. From the menus, select **Mechatronics → Control System → Modify**
 2. In the **Modify Control System** panel, right-click and Browse for the system named:
.MDI_Demo_Vehicle_It_ABS_ESP_01.ABS_system_01.ues_ABS
 3. Set the **Active** to be on
 4. In a similar fashion, turn on the ESP system named:
.MDI_Demo_Vehicle_It_ABS_ESP_01.ESP_system_01.ues_ESP

Function Expression Control System

- **Re-run the lane change simulation**
 1. From the menus, select **Simulate → Full-Vehicle Analysis → Open-Loop Steering Events → Single Lane Change**
 2. Use the same settings as for the last simulation, but change the Output Prefix to be '**with_esp_functions**'
 3. Compare the vehicle yaw rate (**ues_cst_body_yaw_rate_model_units**) between the with- and without-ESP simulations. Also plot the left and right front brake torques (**pt[l,r]_front_brake_torque_data**) for the with-ESP simulation.
- **The effect of the ESP system activating the brakes should be apparent from the yaw rate plot. This ESP system has a target maximum yaw rate of approximately 10 deg/sec; the ESP simulation results should clearly show this.**

Function Expression Control System



Yaw Rate and Brake Torque Comparison, Function-based Controller

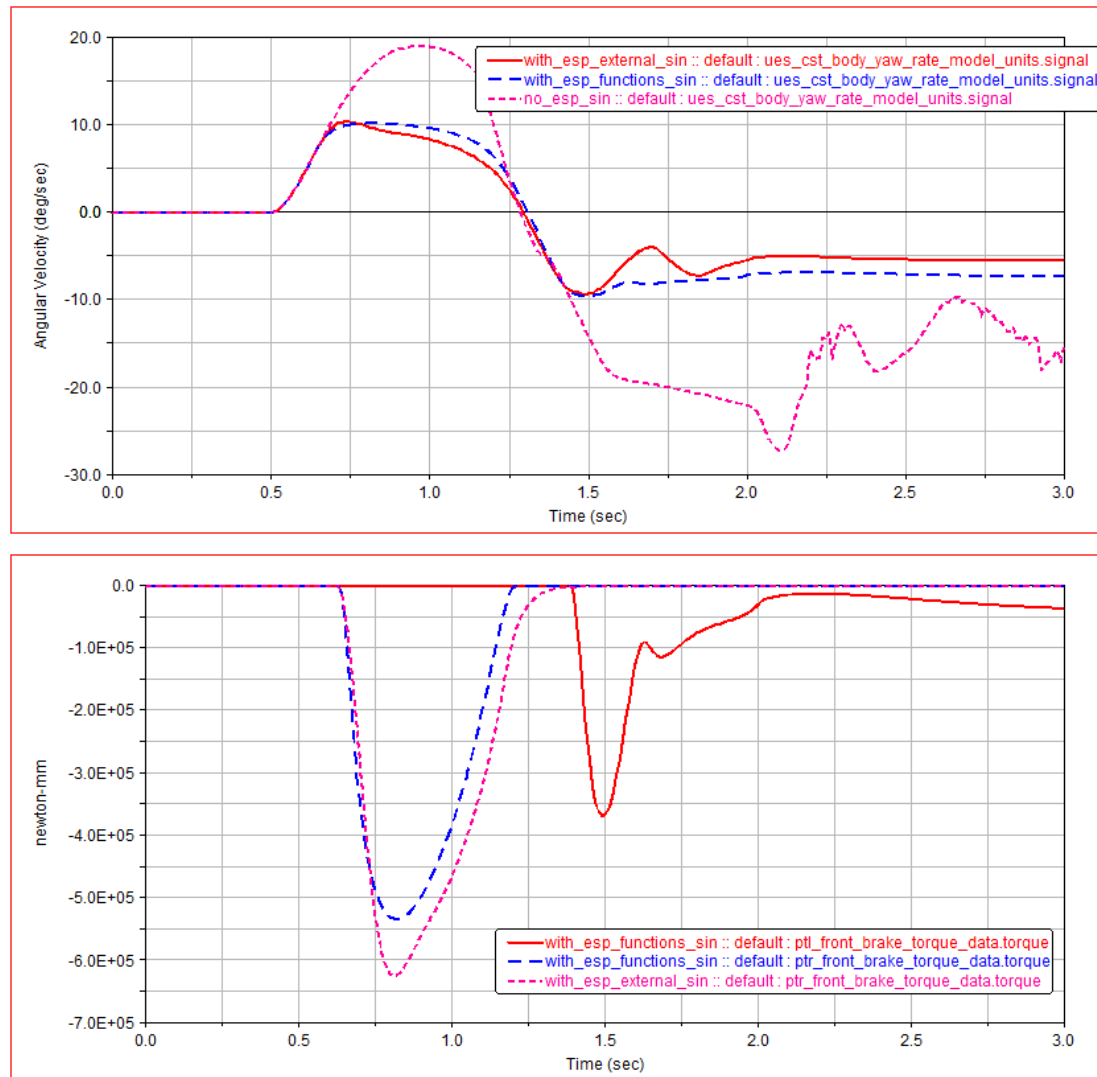
External Control System

- Change 'Function Expression' ABS and ESP Controllers to use controller built externally using Easy5/Matlab.
- Activate the control systems
 1. From the menus, select **Mechatronics** → **Control System** → **Modify**
 2. In the **Modify Control System** panel, right-click and Browse for the system named:
.MDI_Demo_Vehicle_It_ABS_ESP_01.ABS_system_01.ues_ABS
 3. Change System Type to **External System Library**.
 4. For the Library property File use the **ABS_control_system_01.esl** in the **amech_shared/external_system_libraries.tbl** directory
 5. In a similar fashion, turn on the ESP system named:
.MDI_Demo_Vehicle_It_ABS_ESP_01.ESP_system_01.ues_ESP.
 6. For the ESP controller use the **ESP_control_system_01.esl** from the the **amech_shared/external_system_libraries.tbl** directory

External Control System

- **Re-run the lane change simulation**
 1. From the menus, select **Simulate → Full-Vehicle Analysis → Open-Loop Steering Events → Single Lane Change**
 2. Use the same settings as for the last simulation, but change the Output Prefix to be '**with_esp_external**'
 3. Compare the vehicle yaw rate (**ues_cst_body_yaw_rate_model_units**) between the with- and without-ESP simulations. Also plot the left and right front brake torques (**pt[l,r]_front_brake_torque_data**) for the with-ESP simulation.

External Control System



Yaw Rate and Brake Torque Comparison, Imported Library Controller

Mechatronics

- The characteristics of the function-based and external system based controllers are shown to be slightly different. Factors such as different gains/settings used while building the library can affect the results.
- For using the Co-simulation method see the SimCompanion article below:
SimCompanion article #: **KB8017767**
<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8017767>
Using Mechatronics to Incorporate a Vehicle Controller into an Adams/Car Model
- To learn more about building the control system in the template builder go through the tutorial provided in the online help documentation under:
Adams/Car – Mechatronics – Tutorials

APPENDIX H

FOUR-POST VERTICAL EXCITATION TEST

Four-Post Vertical Excitation Test

- **This tutorial teaches you how to work with custom Adams/Car test rigs, custom simulation procedures, and how to create a private Adams/Car binary.**
- **You will work with an existing test rig created in Adams/Car Template Builder. This tutorial will overview the internal workings of the test rig, cover the steps used in making it, and then detail how to edit it and create a custom Adams/Car binary containing the test rig.**



This workshop takes one hour to complete.

Four-Post Vertical Excitation Test

- To go through this tutorial, you need four-post test rig models, macros, and interface files (provided at `C:\MSC.Software\Adams\2017_2\acar\examples\fourpost\analysis`) : `acar_build.cmd`, `acme_4PostRig.cmd`, `macros_ana.cmd`, `mac_ana_ful_fou_sub.cmd` and `fourpost_header.txt`. Ask your instructor for these files.

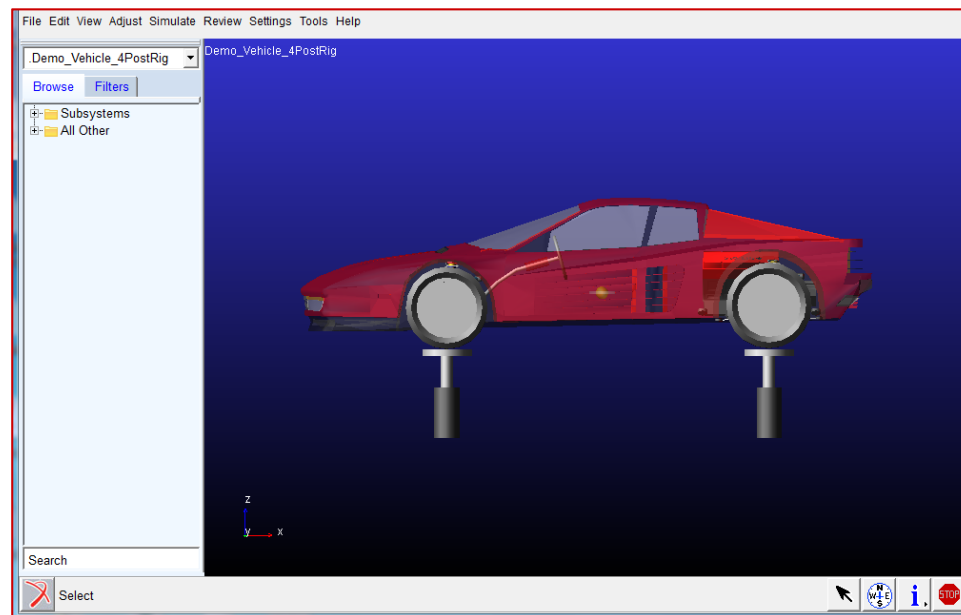
Four-Post Vertical Excitation Test

- **What's in this appendix:**
 - Introduction
 - Creating a test rig template in Adams/Car
 - Preparing to create a private binary
 - Adding custom analysis procedures to Adams/Car

Four-Post Vertical Excitation Test

- **Introduction**

- The objective of this exercise is to investigate the vertical dynamics of the full vehicle and its suspension systems. The test results can be postprocessed in the frequency domain to study the damped natural frequencies of various ride modes and their respective damping levels. Additional insight can also be gathered into the influences of the vehicle's vertical dynamics effects on handling behavior by gaining a further understanding of system dynamic responses including the following:



Four-Post Vertical Excitation Test

- Front to rear modal balance
 - Suspension to body transfer function gain and phase
 - Suspension to tire transfer function gain and phase
 - Tire contact patch vertical load variation
- The test is conducted by assembling a standard full-vehicle model to a special four-post test rig. The test rig is simply defined by four parts representing the tire pads that support the vehicle. These tire pads are constrained to move in only the vertical direction and a displacement actuator (motion controller) controls their vertical motion. The only constraint between the pads and the vehicle's tires is the friction of the tire itself. Because the Delft tire model supports zero velocity tire friction, this is all that is required to constrain the vehicle during the dynamic portion of the simulation.

Four-Post Vertical Excitation Test

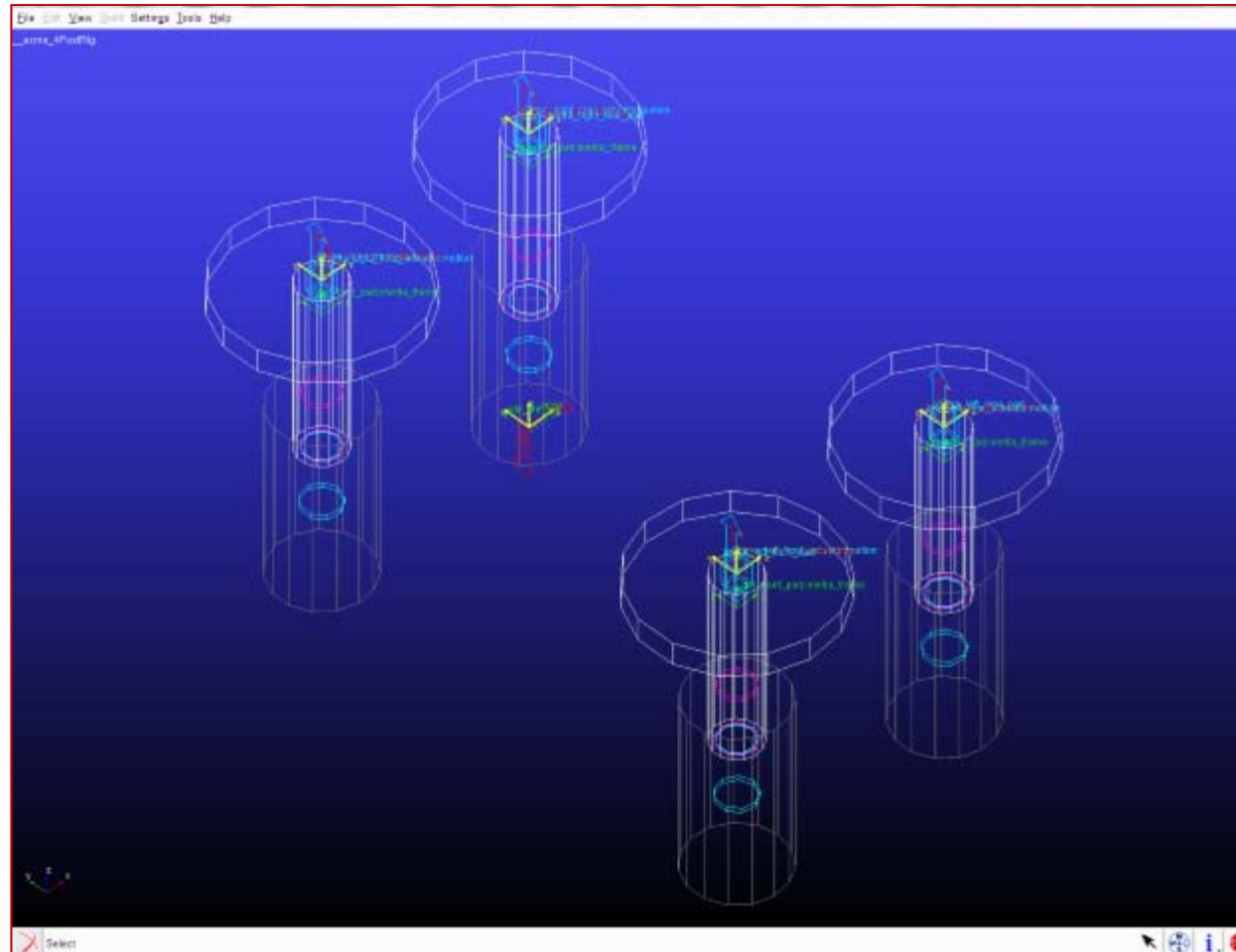
- The vertical actuators are each controlled by an analytical function. The analytical functions are used to describe the displacement profile of the actuator in the time domain and they are limited to constant amplitude sinusoidal input that sweeps over a predetermined frequency range in a set amount of time. When using the analytical function control there exist four excitation modes described below:
 - Heave: all tire pads move vertically in phase.
 - Pitch: the front tire pads move 180 degrees out of phase with the rear tire pads.
 - Roll: the left tire pads move 180 degrees out of phase with the right tire pads.
 - Warp: the left-front and right-rear tire pads move 180 degrees out of phase with the right-front and left-rear pads.

Four-Post Vertical Excitation Test

- **Test rig description**

- The four-post test rig was created in Adams/Car Template Builder and is named **__acme_4PostRig.tpl**. What follows is a brief description of how it works and how it relates to the standard Adams/Car simulation types. The major role of the four-post test rig template is analysis and it contains four general parts. There are four tire pads and four actuators for each of the vertical translation joints on each pad. The location of all of the pads and their respective constraints, actuators, and so on, are parameterized in the ground plane (X and Y) to a wheel center location communicator that comes from the suspension systems. The vertical location is parameterized to the Z location of the **std_tire_ref** marker. The **std_tire_ref** marker has its Z height set automatically during the assembly process so that it represents the vehicles average tire contact patch height.

Four-Post Vertical Excitation Test



Four-Post Vertical Excitation Test

- **User input requirements: simulation description**
 - The analysis input parameters are grouped in two categories: one group of parameters is common to all analyses, while the other group consists of input specific to this four-post test. The four-post simulation input requirements are controlled by the user in order to define the boundary conditions of the desired vertical excitation test.
 - Always required:
 - Output Prefix
 - End Time
 - Number of Steps
 - Type of Analysis (Interactive, Background)
 - Analysis Log File (Yes, No)
 - Four-post simulation input requirements:
 - Peak Displacement
 - Displacement Units (m, mm, inch, and so on)
 - Frequency Range (Units hardcoded to Hz)
 - Excitation Mode (Heave, Pitch, Roll, Warp)

Four-Post Vertical Excitation Test

- **Assembly and simulation process requirements**
 - The following steps outline what the test rig simulation process is doing internally after a simulation has been requested. The vehicle/test rig assembly process is similar regardless of the user input parameters outlined in the previous section of the document. However, the modifications made to the assembled model will vary depending on these parameters.
 - Assembly process:
 1. Same subsystem level check as in any full-vehicle maneuver with possibly an extra check to ensure there are no more than four wheels and two suspensions.
 2. Assemble vehicle with test rig. Location communicators will locate the pads in the X and Y plane.
 3. Loop over the tires and reassign the GFORCE reference markers to the appropriate respective pad. The left front tire's reference marker needs to belong to the **gel_front_pad** part of the test rig, for example.

Four-Post Vertical Excitation Test

4. Assign the Z location of the **std_tire_ref** marker based on the average contact patch location of all of the tires (the same as it is done in a full-vehicle simulation).
5. Set the tire property file to a hardcoded value of **mdi_2d_flat.rdf** for each of the tires without generating any road geometry.
6. Modify the actuator's (**jmf_left_front_actuator**, **jmf_right_front_actuator**, and so on) analytical functions according to the user input data:
 - The following functions need to be assigned to each actuator based on the analytical drive signal:

```
Left Front = LF_phase*Peak_Amplitude*sin(1/2*360D*Freq_Range/End_Time*Time**2)
Right Front = RF_phase*Peak_Amplitude*sin(1/2*360D*Freq_Range/End_Time*Time**2)
Left Rear = LR_phase*Peak_Amplitude*sin(1/2*360D*Freq_Range/End_Time*Time**2)
Right Rear = RR_phase*Peak_Amplitude*sin(1/2*360D*Freq_Range/End_Time*Time**2)
```

Four-Post Vertical Excitation Test

- Where the following values are assigned to the phase variables:
 - Heave Mode: LF_Phase, RF_Phase, LR_Phase, RR_Phase = 1.0
 - Pitch Mode: LF_Phase, RF_Phase = 1.0 & LR_Phase, RR_Phase = -1.0
 - Roll Mode: LF_Phase, LR_Phase = 1.0 & RF_Phase, RR_Phase = -1.0
 - Warp Mode: LF_Phase, RR_Phase = 1.0 & RF_Phase, LR_Phase = -1.0
- The test rig then goes through the following simulation process:
 - Submit the simulation to the solver using a similar process as the full-vehicle simulation. The simulation needs one static equilibrium, an initial velocity = 0.0, and then a dynamic simulation equal to the end. The hmax on the integrator should also be set to at least 1/10 of the maximum frequency range. For example, if the frequency range set by the user is 20Hz then the hmax should be $1/10 \times 1/20 = 1/200$ (0.005). This is necessary to avoid aliasing of the input during the simulation.

Four-Post Vertical Excitation Test

- **Creating a test rig template in Adams/Car**
 - A minimum prerequisite for the task of adding a test rig to Adams/Car is a thorough understanding of all aspects of Adams/Car Template Builder. It is very important that users attempting to create test rigs in Adams/Car have a firm understanding of the concepts of communicators and actuators. As you work through this section, you should reference the Templates tab in the Adams/Car online help. It might also be beneficial to examine the test rigs that are included in standard Adams/Car.
 - A test rig in Adams/Car is almost completely comparable to a template in Adams/Car. Test rigs differ from templates mainly because test rigs contain actuator elements, such as motions and forces, to excite the assembly.

Four-Post Vertical Excitation Test

- The procedure for creating a test rig template in Adams/Car is just like the procedure of creating a normal template, with a few differences. The steps for creating a test rig template are:
 - Creating a test rig template
 - Saving the test rig template
 - Modifying the test rig template
- A test rig template is created in the template builder in Adams/Car. Like a regular template, a test rig template can contain parts attached together via attachments and forces. Unlike most templates, the test rig template will **also contain actuators** to excite the system. The test rig template, like normal templates, will also contain communicators to enable the exchange of information with other templates.

Four-Post Vertical Excitation Test

- Because templates and test rigs are so similar, it would be redundant to fully describe how to create test rigs. Instead, for specific information about building templates and communicators, see the Templates tab in the Adams/Car online help.
- Adams/Car works with test rigs as templates. However, in order to incorporate a test rig for use on an assembly, the test rig must be converted to a test rig model file (.cmd) and a private Adams/Car binary created. You can, of course, create a new test rig template from the Adams/Car interface very easily, but it is often best to work with existing templates in order to better understand the capabilities of Adams 2017.2.
- You start by modifying an existing test rig model file (.cmd) in the template builder. Start by locating the file **acme_4PostRig.cmd**. This is a test rig model file (.cmd) that contains a test rig currently equipped with three active posts.

Four-Post Vertical Excitation Test

- **Loading a test rig model file (.cmd) into the Adams/Car Template Builder**
 - If you want to open an existing test rig model (.cmd) file and edit it using Adams/Car Template Builder, follow the steps outlined in this section. Try these steps out on the **acme_4PostRig.cmd** file. Note that you should be careful if using a private or site Adams/Car binary when editing a test rig template. If you already made a test rig and a private binary incorporating it, you cannot edit the test rig of the same name.

Four-Post Vertical Excitation Test

- To load a test rig model file:

1. Move the test rig model file, **acme_4PostRig.cmd** file, to your private car database under the templates table (for example, C:\private.cdb\templates.tbl\).
2. Rename the file from **acme_4PostRig.cmd** to **__acme_4PostRig.tpl**. Make sure that you precede the template name with two underscores (__).
3. Open the file in a text editor.
4. Insert the header as follows:
 - Note that **TEMPLATE_NAME** must match the file name exactly (excluding the extension).

```
$-----MDI_HEADER
[MDI_HEADER]
FILE_TYPE      = 'tpl'
FILE_VERSION   = 2017.2
FILE_FORMAT    = 'ASCII'
HEADER_SIZE    = 9
(COMMENTS)
{comment_string}
'Adams Car analysis template'
```

Four-Post Vertical Excitation Test

```
$-----TEMPLATE_HEADER  
[TEMPLATE_HEADER]  
TEMPLATE_NAME = '__acme_4PostRig'  
MAJOR_ROLE    = 'analysis'  
TIMESTAMP     = '2017/07/07,19:53:00'  
HEADER_SIZE   = 6
```

- Tip: You can copy the text from the file **four_post_header.txt**, located in **four_post_test_rig_files**, and paste it into your file.
5. There is a variable inside the **acme_4postrig.cmd** file called **model_class**. Search for **model_class**. Change this variable assignment to look like the following (this variable must be set to template for Template Builder to use it):

```
variable create &  
variable_name = __acme_4PostRig.model_class &  
string_value = "template" &  
comments = "Memory for ADAMS/Car model class"  
!
```

6. Save the file.

Four-Post Vertical Excitation Test

- **Opening the test rig in Adams/Car Template Builder**
 - The file that you just specified as an Adams/Car template contains a model of a test rig with only two active posts. In this section, you activate the remaining two (left and right-front) posts.
- **To open the test rig:**
 1. Make sure that your **.acar.cfg** file specifies you as an expert user so you can start the Adams/Car Template Builder.
 2. Start Adams/Car and select the template-builder mode at the prompt.
 3. From the **File** menu, select **Open**.
 4. Right-click the Template Name text box, point to **Search**, and then select **<private>/ templates.tbl**.
 5. Select the **__acme_4PostRig.tpl** template file, and then select **OK** twice.
 6. Make sure icon visibility is **on**, the view is set to **Front Iso** and the view is zoomed to fit.

Four-Post Vertical Excitation Test

- **Modifying the test rig in Adams/Car Template Builder**
 - To make the actuator active, you will add a variety of items to the test rig: a bumpstop, a reboundstop, a joint attachment, a joint motion actuator, and a joint force actuator.
- **To add a bumpstop:**
 1. From the **Build** menu, point to **Forces**, point to **Bumpstop**, and then select **New**.
 2. In the **Bumpstop** Name text box, enter **front_extension_stop**.
 3. Specify the **I Part** as **gel_front_pad** by typing it in or by right-clicking the text box, pointing to Part, selecting **Pick**, and then choosing the part **gel_front_pad**.
 4. Specify the **J Part** as **ground** by right-clicking the text box, pointing to Guesses, and then selecting ground.

Four-Post Vertical Excitation Test

5. Specify the **I Coordinate Reference** as **cfl_Front_Actuator_Base**.
6. Specify the **J Coordinate Reference** as **cfl_Front_Pad**.
7. Ensure that the Property File text box specifies
<shared>\bumpstops.tbl\mdi_0001.bum as the property file.
8. In the **Clearance** text box, enter **127**.
9. Select **OK**.

Note that because of symmetry relations, not only is a bumpstop immediately created on the left front actuator base, but one is also created on the right front.

Four-Post Vertical Excitation Test

- **To create a reboundstop:**

1. From the Build menu, point to Forces, point to Reboundstop, and then select **New**.
2. In the Reboundstop Name text box, enter **front_retract_stop** to enforce a consistent naming convention.
3. Set I Part to **gel_front_pad**.
4. Set J Part to **ground**.
5. Set I Coordinate Reference to **cfl_Front_Actuator_Base**.
6. Set J Coordinate Reference to **cfl_front_pad**.
7. Ensure that the Property File text box points to **<shared>\reboundstops.tbl\mdi_0001.reb**.
8. In the Clearance text box, enter **127**.
9. Select **OK**.

Note that because of symmetry relations, not only is a reboundstop immediately created on the left front actuator base, but one is also created on the right front.

Four-Post Vertical Excitation Test

- To add an attachment between the actuator pad and actuator base:

1. From the Build menu, point to Attachments, point to Joint, and then select **New**.
2. Specify the Joint Name as **left_front_pad**.
3. Specify the I Part as **ground**.
4. Specify the J Part as **gel_front_pad**.

Note: The I and J part order is important to determine direction for the joint motion actuator defined next.

5. Set the Type to **single**.
6. Set Joint Type to **translational**.

Four-Post Vertical Excitation Test

7. Set Location Dependency to **Delta location** from coordinate.
8. Set Location Coordinate Reference to **cfl_front_pad**.
9. Set Location to **0,0,0** in local.
10. Set Orientation Dependency to **Delta orientation** from coordinate.
11. Set Construction Frame to **cfl_front_pad**.
12. Set Orientation to **0,0,0**.
13. Select **OK**.

Four-Post Vertical Excitation Test

- **To make the joint motion actuator:**
 1. From the Build menu, point to Actuators, point to Joint Motion, and then select **New**.
 2. Specify the Actuator Name as **left_front_actuator** to enforce consistent naming.
 3. Specify the joint as **jostra_left_front_pad** by picking the joint you just created.
 4. In the Application text box, enter **pad_excitation**.
 5. In the Identifier text box, enter **left_front_actuator**.
 6. In the Function text box, enter **15*sin(720d*time)**. This is a dummy function that the analysis macro will overwrite.
 7. Specify the Time Derivative as **displacement**.

Four-Post Vertical Excitation Test

8. Set the Force Limits to **-5e4,5e4**.
9. Set the Displacement Limits to **-1000,1000**.
10. Set the Velocity Limits to **-1e4,1e4**.
11. Set the Acceleration Limits to **-9.8e4,9.8e4**.
12. In the Units text box, enter **mm**.
13. Select **OK**.

Four-Post Vertical Excitation Test

- **To make the joint force actuator:**
 1. From the Build menu, point to Actuators, point to Joint Force, and then select **New**.
 2. Specify the Actuator Name as **If_force_actuator**.
 3. Pick the Joint as **jostra_left_front_pad**.
 4. In the Application text box, enter **Left front actuator force**.
 5. In the Identifier text box, enter **Left front actuator force**.
 6. In the Function text box, right-click and point to **Function Builder**.
 7. **Paste** the following into the text box:

```
(STEP(TIME,0,1,0.002 0)*  
(1e5)*VARVAL(If_actuator_disp))  
+STEP(VARVAL(If_actuator_vel),-pvs_Friction_Saturation_Velocity,  
-pvs_Friction_Saturation_Force,pvs_Friction_Saturation_Velocity  
,pvs_Friction_Saturation_Force)  
-VARVAL(If_actuator_force)
```

8. Select **OK**.

Four-Post Vertical Excitation Test

- **Saving the test rig template**
 - The test rig template is saved to a file just like a regular template. The test rig template should be saved in an ASCII format to facilitate the modifications that are required and described in the next section.
- **To save the template:**
 1. From the **File** menu, select **Save As**.
 2. Make sure the file format is set to **ASCII**, and that **Zero Adams Ids** is selected.
 3. Select **Close Template**.
 4. Select **OK**, don't save a backup copy when prompted.

Four-Post Vertical Excitation Test

- **Making a test rig model file (.cmd) from an Adams/Car template**
 - You must manually modify the file of the test rig template to make it into a test rig model file (.cmd). There are two modifications that you must do to the ASCII template file generated from Adams/Car.
 - If you want to take a test rig built in the template builder and then use it as a test rig, you should basically perform the steps in Loading a test rig model file (.cmd) into the Adams/Car Template Builder, on slide 18 in reverse order.

Four-Post Vertical Excitation Test

- **To make a test rig from a template:**
 1. Outside of Adams/Car, from your private database, copy the file **__acme_4PostRig.tpl** from the **templates.tbl** table to your private directory (**C:\acar_private**) or another directory.
 2. Rename the file to **acme_4PostRig.cmd**.
 3. Using a text editor, open **acme_4PostRig.cmd**.
 4. There is a variable inside **__acme_4PostRig.tpl** named **model_class**. Change this variable assignment to look like the following (this variable must be set to test rig in order for it to be used as one):

```
variable create &  
  variable_name = __acme_4PostRig.model_class &  
  string_value = "testrig" &  
  comments = "Memory for Adams/Car model class"  
!
```

Four-Post Vertical Excitation Test

5. Remove the header information that is added at the beginning of the ASCII template file. This header must be removed because the command file reader will not understand the information stored in this header and will output errors. A typical header from an ASCII was shown in Loading a test rig model file (.cmd) into the Adams/Car Template Builder, on slide 18.
6. Remove all the lines from the beginning of the file up to and including the line containing the HEADER_SIZE attribute.
7. Save the file.

Four-Post Vertical Excitation Test

- **Adams/View variables required in a test rig**
 - Templates and test rigs in Adams/Car have information that is stored in Adams/View variables to determine how the template is used. All templates, including test rigs have three required variables: major role, minor role, and model class. Test rigs have an additional Adams/View variable named assembly class.
 - All the variables required in a test rig are described below. The first three variables: role, minor_role, and model_class are all created automatically when the test rig template is created. You must manually create the variable testrig_class, which is unique to test rigs, as described above.

Four-Post Vertical Excitation Test

- **Major role**
 - The major role of Adams/Car templates and test rigs is stored in an Adams/View variable named role. The major role of a test rig is always analysis. When creating a test rig in Adams/Car, it is important to ensure that this value is set properly.

```
variable create &  
  variable_name = __acme_4PostRig.role &  
  string_value = "analysis" &  
  comments = "Memory for Adams/Car major role"
```

Four-Post Vertical Excitation Test

- **Minor role**

- The minor role of Adams/Car templates and test rigs is stored in an Adams/View variable named `minor_role`. The minor role of a test rig is typically any. Setting the minor role to any is extremely important if you are designing a test rig that is supposed to work with other subsystems that can have different minor roles. For example, a suspension test rig should work with either front, rear, or trailer type suspensions. If the minor role of the test rig is defined as any in this case, the communicators will hook up properly between the test rig and suspension subsystem.

```
variable create &  
  variable_name = __acme_4PostRig.minor_role &  
  string_value = "any" &  
  comments = "Memory for Adams/Car minor role"
```

Four-Post Vertical Excitation Test

- ***Model class***

- Every model in Adams/Car has a specific model class. The model class of a model is stored in an Adams/View variable named model_class. This variable is automatically created at the time the model is created. Currently, in Adams/Car, there are four model classes defined: template, subsystem, test rig, and assembly.

```
variable create &  
  variable_name = __acme_4PostRig.model_class &  
  string_value = "testrig" &  
  comments = "Memory for Adams/Car model class"  
!
```

Four-Post Vertical Excitation Test

- ***Test rig class***

- All test rigs in Adams/Car can be associated with a particular class of assembly. For example, the __MDI_SUSPENSION_TESTRIG test rig is associated with suspension assemblies. The test rig class of a test rig is stored in an Adams/View variable called testrig_class.

```
variable create &  
variable_name = __acme_4PostRig.testrig_class &  
string_value = "full_vehicle" &  
comments = "Memory for Adams/Car testrig class"
```

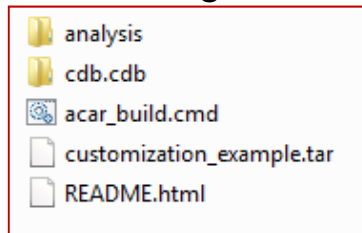
- The testrig_class variable is used directly in the graphical user interface. For example, this variable is used in the new suspension assembly and new full-vehicle assembly dialog boxes. These two dialog boxes each contain an option menu from which you select the test rig to be included in the new assembly. This option menu will only contain test rigs that are compatible with that class of assembly. The option menu on the new suspension assembly dialog box, for example, will only list those test rigs that have a test rig class of suspension.

Four-Post Vertical Excitation Test

- **Preparing to create a private binary**

- You can now add your new test rig model file (acme_4PostRig.cmd) and the macro files attached to this tutorial to build a custom private Adams/Car binary which can implement this new test rig. The process of creating private and site binary files is outlined next:

1. If the directory does not exist, create a directory **C:\acar_private** (Adams 2017.2 always looks for **\$HOME\acar_private** when running a private binary, unless **MDI_ACAR_PRIVATE** explicitly sets this otherwise).
2. In this directory, copy the file **customization_example.tar** from the path **C:\MSC.Software\Adams\2017_2\acar\examples\fourpost** into **C:\acar_private**.
3. Extract the customization_example.tar to get all the files required for creating private binary, it will take out the following files and folders:



4. Replace the **acme_4PostRig.cmd** file existing in path: **"C:\acar_private\analysis\models"** with the **acme_4PostRig.cmd** file you created in step 32.

Four-Post Vertical Excitation Test

- A description of each of these follows:

acar_build.cmd

- This is the file upon which Adams 2017.2 will call when building a private binary. In general, this file contains any commands to:
 - modify the Adams 2017.2 car interface
 - import the test rig model files
 - add libraries (which show up in the command navigator)
 - add macros
 - as well as some standard commands which should be in any **acar_build.cmd** file
- The following is the text of the **acar_build.cmd** file:

```
!---- Create custom libraries for storage ----  
library create library_name=.ACME  
library create library_name=.ACME.macros
```

- The two lines above create special storage for all the acme macros that will be created.

Four-Post Vertical Excitation Test

acme_4PostRig.cmd

- This file contains the test rig model file (.cmd) that you just created. This test rig model will be imported into your private binary and now be available to any future assemblies.

macros_ana.cmd

- This file serves as a pointer to mac_ana_ful_fou_sub.cmd. It contains a hard-coded file location. It is good practice to use pointers like this rather than to import the simulation macros themselves. This allows for easy modification.

Edit this file to make sure the commands are appropriate to your computer.

mac_ana_ful_fou_sub.cmd

- This is a macro that instructs Adams/Car on how to simulate the model.

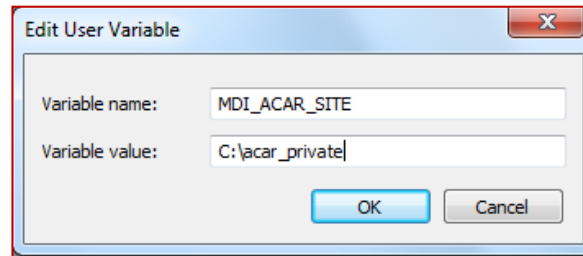
dboxes_ana.cmd

- This file serves the following functions:
 - Checks if any dialog box is exists and delete it if exist.
 - Reads four post shaker dialog box
 - Adds menu option

Four-Post Vertical Excitation Test

- **Set environment variable:**

- Set the environment variable MDI_ACAR_SITE to the location where your file are stored, here C:\acar_private.



- **Build the Adams/Car site binary file as follows:**

- On **Windows**: From the Start menu, point to Programs, point to **MSC.Software**, point to **Adams 2017.2**, and then select **Adams - Command**. In the command window, type **acar; cr-sitebin**
- On **UNIX**: Type **adams2017.2**; type **acar**, and then type **cr-sitebin**

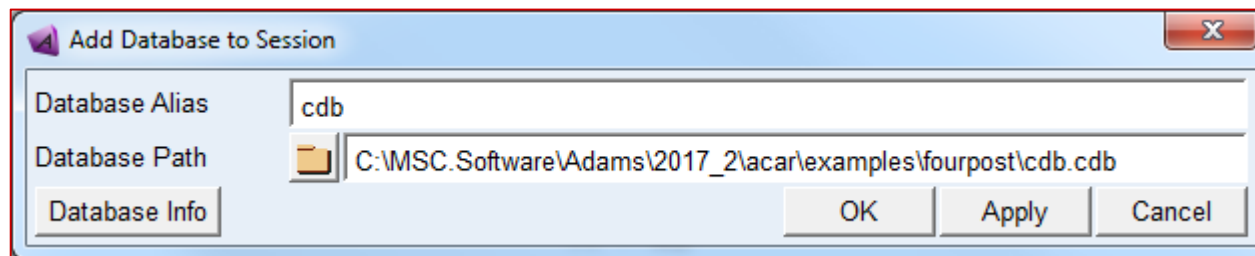
- **Adams/Car automatically executes acar_build.cmd and builds all the appropriate entities and libraries into a site version of the Adams binary file.**

Four-Post Vertical Excitation Test

- **Run the Adams/Car site binary file:**
 - On **Windows**: From the Start menu, point to Programs, point to **MSC.Software**, point to **Adams 2017.2**, and then select **Adams - Command**. In the command window, type **acar; ru-site**
 - On **UNIX**: Type **adams2017.2**; type **acar**, and then type **ru-site**
- Check the acar.log file for information/errors on building the **acar.bin** file.

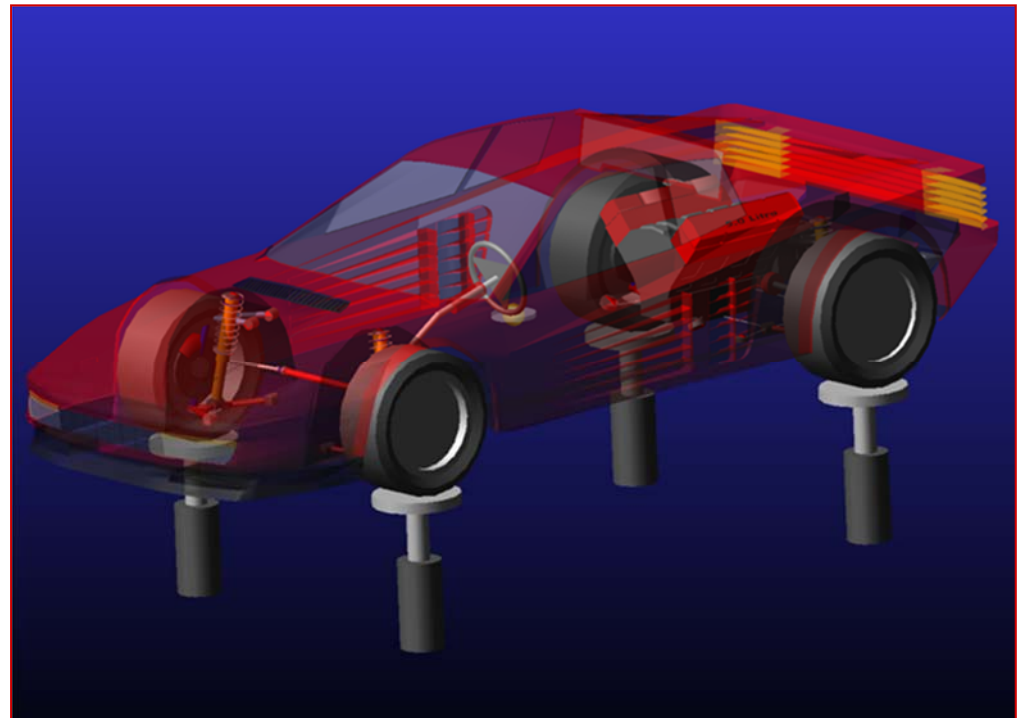
Four-Post Vertical Excitation Test

- **Loading an Assembly with the Test Rig**
 - Once you have successfully built the site binary file and started the site version of Adams/Car, you must open the assembly that we have also provided in the examples directory (cdb.cdb).
 - If you have not already added the database to your .acar.cfg file, you should do the following:
 - From the Tools menu, point to Database Management, and then select Add to Session.
- **Open the assembly contained in the cdb database as follows:**
 1. From the **File** menu, point to **Open**, and then select **Assembly**.
 2. Open the **MDI_Vehicle_4PostRig.asy**.



Four-Post Vertical Excitation Test

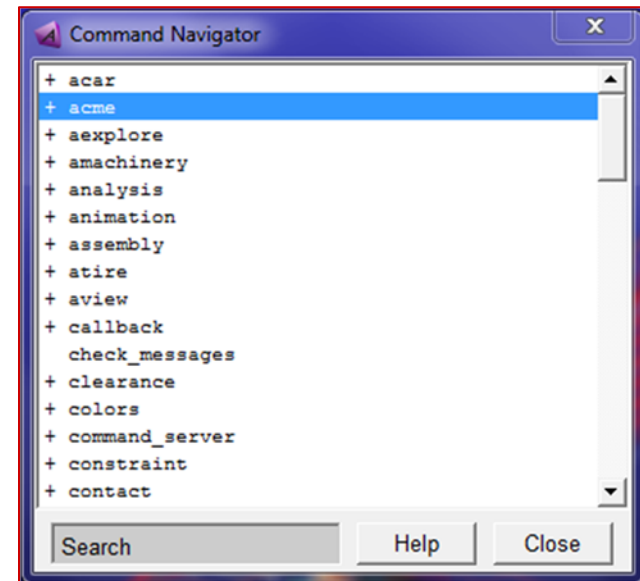
- You can also create a new assembly by doing the following:
 - From the **File** menu, point to **New**, and then select **Full-Vehicle Assembly**.
 - Select the appropriate subsystem files.
 - Change the vehicle test rig from **__MDI_SDI_TESTRIG** to **__acme_4PostRig**.
 - Select **OK**.



Four-Post Vertical Excitation Test

- **Running a Test:**

- Use the Command Navigator to submit an analysis using the auto-generated dialog box:
- To open the Command Navigator, from the Tools menu, select **Command Navigator**.
- To run a full-vehicle analysis with the four-post test rig, you must select **acme -> analysis -> full_vehicle -> four_post -> submit** (double-click).
- Adams/Car displays the following dialog box (without the values, which we added for your convenience):



Four-Post Vertical Excitation Test

3. After the analysis is completed, review the results using the animation and post-processing tools.

Parameter	Value
Assembly	.__acme_4PostRig
Output Prefix	example
Comment	An example fourpost testrig analysis
End Time	8
Number Of Steps	800
Peak Displacement	10
Units	mm
Frequency Range	5
Excitation Mode	heave
Analysis Mode	interactive
Load Results	yes
Log File	yes
Error Variable	.ACAR.variables.errorFlag

OK Apply Cancel

Four-Post Vertical Excitation Test

- **Adding custom analysis procedures to Adams/Car**
 - A minimum prerequisite of adding an analysis to Adams/Car is a thorough understanding of the Adams/View macro language. As you work through this section, you should reference the Customize tab in the Adams/Car online help.
 - Adams/Car is designed with an open architecture to facilitate extensions based on customer requirements. Most often users desire to extend the analysis capabilities of Adams/Car to include an analysis type or maneuver that is typical within their company, but not included in the Adams/Car product. All analyses in Adams/Car are defined using macros. Therefore, adding a new analysis to Adams/Car can be as simple as adding a new macro. A custom graphical interface to the new macro may also be created to facilitate usage, but is not required.
 - This example will provide you with step-by-step instructions to create your own macro from a given user scenario.

Four-Post Vertical Excitation Test

- The best way to get started creating a new analysis is not to start from scratch. There is a broad range of analyses that are possible within the Adams/Car product as it is shipped. Examine the existing functionality to find an analysis that closely suites the analysis you want. Once you have selected an existing macro, simply modify it to suite your needs.
- You can view any of the standard macros using the macro editor inside Adams/Car. From the Adams/Car menu, choose Tools -> Macro -> Edit -> Modify to open the Database Navigator. Then, select the macro you want to display. Selecting mac_ana_ful_fou_sub from the Database Navigator, for example, will allow you to view the standard macro for submitting a full-vehicle four-post analysis.



For more information on macros in Adams/Car, see the Customize tab in the Adams/Car online help.

Four-Post Vertical Excitation Test

- Where possible, existing utility macros should be used in custom analysis macros. Using the existing utilities macros shipped within Adams/Car has a few benefits. First, these macros are already written and tested by MSC.Software. Second, using these macros reduces the amount of work required to add a custom analysis to Adams/Car. Third, these macros provide a common basis for customization, facilitating maintenance and debugging between different projects.
- **Analysis macro structure**
 - In general, all the analysis macros within Adams/Car are structured the same. This is especially true if you are looking only at a particular class of analysis macros. For example, all the open-loop full-vehicle analysis macros are surprisingly similar. The same holds true for all the full-vehicle analysis macros based on the Driving Machine. Typically, a small section within a macro is what makes that macro and the resulting analysis unique. Every analysis macro in Adams/Car has the same basic structure.

Four-Post Vertical Excitation Test

- **Parameter definition**

- Parameters are the means by which the end user inputs values into the macro. They are placeholders for information that the end user provides when the macro is executed.

Macro parameters are described in detail in the Adams/View online help.

- **Parameters common to all analyses:**

- Output prefix
- End time
- Number of steps
- Type of analysis (interactive, background)
- Analysis log file (yes, no)

The assembly parameter is used to specify which Adams/Car assembly within the session will be analyzed. The output_prefix parameter is used to specify the unique prefix to be added when creating output files specific to the analysis. These parameters are usually defined first in the analysis macro as seen in the example.

Four-Post Vertical Excitation Test

- **Parameters specific to this four-post test rig:**

You use the four-post simulation input parameters to define the boundary conditions of the desired vertical excitation test. The parameters are:

- Peak displacement
- Displacement units (such as m, mm, inch)
- Frequency range (units hardcoded to Hz)
- Excitation mode (heave, pitch, roll, or warp)

```
! $assembly:t=model
! $output_prefix:t=string
! $comment:t=string:d= " "
! $end_time:t=real:gt=0
! $number_of_steps:t=integer:gt=0
! $analysis_mode:t=list(interactive,graphical,background):d=interactive
! $peak_displacement:t=real:gt=0
! $units:t=list(mm):d=mm
! $frequency_range:t=real:gt=0
! $excitation_mode:t=list(heave,pitch,roll,warp):d=heave
! $load_results:t=list(yes,no):u=yes
! $brief:t=list(on,off):u=off
! $log_file:t=list(yes,no):u=yes
! $error_variable:t=variable:d=.ACAR.variables.errorFlag
```

Four-Post Vertical Excitation Test

- **Parameter descriptions:**

- **assembly:** This parameter expects the user to specify an existing model.
- **analysis_name:** A string value indicating the name prefix for all files common to this analysis.
- **end_time:** A real value telling Adams/Solver the end time of the four-post maneuver.
- **number_of_steps:** An integer value telling Adams/Solver the number of output steps.
- **analysis_mode:** A string value to specify the mode of the analysis. The two valid modes are interactive or background.
- **peak_displacement:** The maximum amplitude of the shaker pad vertical displacement.
- **units:** Hardcoded to mm for tutorial, can be expanded to include other units.
- **frequency_range:** A real value indicating the frequency range of the actuator motion functions.
- **excitation_mode:** A list value indicating the direction of the shaker pad motions.
- **log_file:** Indicates if an analysis log file should be generated.

Four-Post Vertical Excitation Test

- **Error handling**

- The error handling section of each analysis macro should contain checks to identify inappropriate assemblies or invalid user-entered values. The error handling section should also contain checks to ensure the assembly contains subsystems that are required to perform a desired maneuver. For example, if you are creating an analysis macro to perform a full-vehicle, straight-line braking maneuver, a check should be performed to ensure that a brake subsystem exists within the assembly. The error handling section should also have checks to ensure that the values specified by the user for specific parameters are realistic.
- The four-post analysis must be performed with the `__acme_4PostRig` test rig you have created earlier. The setup of the assembly and test rig, described in a later section, perform actions based on the elements known to exist in the `__acme_4PostRig` test rig. In addition to verifying that the correct test rig is used, the macro also determines if the analysis name is unique for this assembly.

Four-Post Vertical Excitation Test

```
variable set variable_name=$error_variable integer=0
!---- Check to ensure the assembly has the proper testrig ----
if condition=($assembly.original_testrig_name != "__acme_4PostRig")
    acar toolkit warning &
    w="Analysis cannot be submitted!", &
    "The assembly does not have the proper testrig. This analysis only", &
    "works with assemblies using the '__acme_4PostRig' testrig."
    variable set variable_name=$error_variable integer=1
    return
end
!---- Check if analysis name already exists ----
if condition=(db_exists("$assembly.$"output_prefix"_fourpost"))
    if condition=(alert("question","An analysis called
\\$'output_prefix'_fourpost\ already exists. Overwrite it?","Yes","No","",2) == 2)
        variable set variable_name=$error_variable integer=1
        return
    end
end
end
```

- The acar toolkit warning utility macro used in the above example will display the user message in a small window that will remain open until you select OK. This utility macro is used throughout Adams/Car to inform the user they have entered an error condition.

Four-Post Vertical Excitation Test

- **Reading property files**

- Many force elements within Adams/Car get their nonlinear characteristics from property files. This information must be updated immediately before an analysis is performed. If the property files are not read, the force elements will typically contain inappropriate information that will directly affect the accuracy of the analysis results.
- After validation of the assembly, property file information must be read in and assigned. The following example can be used directly within a user's analysis macro. The \$assembly parameter must be an Adams/Car assembly as described above. Property files are read in via a user function as described in the following example. This example also demonstrates how to use the acar toolkit message utility macro to descriptive text to the message window. It is very important that the property files are read before the analysis is submitted to Adams/Solver.

Four-Post Vertical Excitation Test

- **Setting up assembly and test rig**

```
!---- Clear out message window ----  
acar toolkit message &  
message=" " append=no display=no closeable=yes  
echo_to_logfile=no  
!---- Read property files ----  
acar toolkit read property_files &  
assembly=$assembly &  
verbose=yes &  
error_variable=$error_variable  
if condition=($error_variable != 0)  
return  
end
```

- The setup of the assembly and test rig is the section of the analysis macro that is unique from other analysis macros. Within this section of the macro, the macro modifies elements of the test rig and assembly based on the type of maneuver being performed and the parameters specified by the user.

Four-Post Vertical Excitation Test

- The setup of the assembly and test rig is the section of the four-post analysis macro that is unique from other analysis macros. Within this section of this macro, elements of the test rig and assembly are modified specific to the four-post maneuver and user input. The code fragments for the four-post setup are shown below, with a description for each section where needed.

```
!---- Setup the assembly for the maneuver ----  
acar toolkit message &  
message="Setting up vehicle assembly for four post shaker..."
```

- The tire reference markers need to be assigned to the appropriate test pad on the shaker table. The naming conventions of the communicator variables for the reference markers are considered fixed, in that the macro looks for the communicators known to exist in the four-post test rig. Note that the setup of the tire reference markers will only occur once for a particular assembly; if the same assembly is used for multiple four-post analyses, the initial setup will be valid for each analysis.

Four-Post Vertical Excitation Test

- For each wheel, the tire reference marker is assigned to a shaker pad. The first step is to find each tire in the full-vehicle assembly. The reassignment occurs via an output communicator in the test rig.



For more information on communicators, see the Templates tab in the Adams/Car online help.

- The communicator holds the name of the part on the shaker pad where the tire reference marker should be attached.

```
if condition=(!db_exists("$assembly.fourpostSetup"))
!---- Parameterize the 4post pad height to the global road height marker just previously adjusted ----
variable set variable=$_self.frontWheel &
object_value=(eval(subsystem_lookup($assembly,"wheel","front")))
variable set variable=$_self.leftFrontWheel &
object_value=(eval(db_filter_name(db_children($_self.frontWheel[1],"ac_tire"),"til_*"))
variable set variable=$_self.rightFrontWheel &
object_value=(eval(db_filter_name(db_children($_self.frontWheel[1],"ac_tire"),"tir_*"))
variable set variable=$_self.rearWheel &
object_value=(eval(subsystem_lookup($assembly,"wheel","rear")))
variable set variable=$_self.leftRearWheel &
object_value=(eval(db_filter_name(db_children($_self.rearWheel[1],"ac_tire"),"til_*"))
variable set variable=$_self.rightRearWheel &
object_value=(eval(db_filter_name(db_children($_self.rearWheel[1],"ac_tire"),"tir_*"))
marker modify &
marker_name=(eval($_self.leftFrontWheel.object_value.ref_marker.object_value)) &
new_marker_name=(eval($assembly.testrig.col_front_pad_mount[1]//".//"
$_self.leftFrontWheel.object_value.ref_marker.object_value.name))
```

Four-Post Vertical Excitation Test

```
marker modify &
marker_name=(eval($_self.rightFrontWheel.object_value.ref_marker.object_value)) &
new_marker_name=(eval($assembly.testrig.cor_front_pad_mount[1]//". " //
$_self.rightFrontWheel.object_value.ref_marker.object_value.name))
marker modify &
marker_name=(eval($_self.leftRearWheel.object_value.ref_marker.object_value)) &
new_marker_name=(eval($assembly.testrig.col_rear_pad_mount[1]//". " //
$_self.leftRearWheel.object_value.ref_marker.object_value.name))
marker modify &
marker_name=(eval($_self.rightRearWheel.object_value.ref_marker.object_value)) &
new_marker_name=(eval($assembly.testrig.cor_rear_pad_mount[1]//". " //
$_self.rightRearWheel.object_value.ref_marker.object_value.name))
variable set variable=$assembly.fourpostSetup &
integer_value=1
end
```

Four-Post Vertical Excitation Test

- The motion actuators driving the displacement of the shaker pads must be reset for each individual four-post analysis. This is in contrast to the tire reference marker setup described above, which needs to occur only once for a particular assembly, and remains valid for all successive four-post analyses. The four excitation modes are heave, pitch, roll, and warp. Each of the four shaker pads will have the same magnitude of motion, but a specific excitation mode will determine the direction of the motion. In heave mode, all four shaker pads will move in the same direction. In pitch mode, the front and rear tires will move in opposite directions. For roll mode, the left and right tires have motion in opposite directions. When warp mode is specified, the left front and right rear tires will move opposite to the direction traveled by the right front and left rear tires. The different excitation modes are achieved by specifying a "1" or "-1" multiplier at the beginning of the actuator function definition.

Four-Post Vertical Excitation Test

```
!---- Assign actuator functions based on excitation mode ----
!-Heave Excitation
if condition=("$excitation_mode" == "heave")
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_front_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_front_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_rear_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_rear_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
!- Pitch Excitation
elseif condition=("$excitation_mode" == "pitch")
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_front_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_front_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_rear_actuator &

function="-1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_rear_actuator &

function="-1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
!- Roll Excitation
elseif condition=("$excitation_mode" == "roll")
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_front_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_front_actuator &
```

Four-Post Vertical Excitation Test

```
function="-1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_rear_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_rear_actuator &

function="-1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    !- Warp Excitation
    elseif condition=("$excitation_mode" == "warp")
        acar template_builder actuator set function &
        actuator=$assembly.testrig.jms_left_front_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_front_actuator &

function="-1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_left_rear_actuator &

function="-1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    acar template_builder actuator set function &
    actuator=$assembly.testrig.jms_right_rear_actuator &

function="1*$peak_displacement*sin(.5*360d*$frequency_range/$end_time*time**2)"
    end
```

Four-Post Vertical Excitation Test

- **Submitting the analysis**

- A common requirement of submitting the analysis to Adams/Solver is the existence of an Adams dataset file (.adm) and an Adams command file (.acf). The two basic types of analyses within Adams/Car are suspension and full-vehicle analyses. In both of these cases, a utility macro is used to generate the ADM and ACF files and submit the analysis to Adams/Solver. These utility macros are executed from within each of the suspension or full-vehicle analysis macros.
- After setting up the assembly, it is ready to be submitted for the four-post analysis. Since this is a full-vehicle assembly, the corresponding full-vehicle submission utility macro is used. In this case, two additional parameters are specified to have non-default values for the four-post simulation: `generate_road_geometry` and `simulation_type`.

Four-Post Vertical Excitation Test

- The code fragment for calling the four-post full-vehicle submission macro is shown, with the important associated parameters described next.

```
!---- Perform the analysis ----  
acme analysis full_vehicle four_post submit &  
  assembly=$assembly &  
  analysis_name="$'output_prefix'_fourpost" &  
  end_time=$end_time &  
  number_of_steps=$number_of_steps &  
  analysis_mode=$analysis_mode &  
  load_results=$load_results &  
  brief=$brief &  
  road_data_file="BEDPLATE" &  
  generate_road_geometry=no &  
  simulation_type=fourpost
```

Four-Post Vertical Excitation Test

– Parameter descriptions:

- **assembly:** This parameter expects the user to specify an existing model.
- **analysis_name:** A string value indicating the name prefix for all files common to this analysis.
- **end_time:** A real value telling Adams/Solver the end time of the maneuver.
- **number_of_steps:** An integer value telling Adams/Solver the number of output steps.
- **analysis_mode:** A string value to specify the mode of the analysis. The two valid modes are interactive or background.
- **load_results:** A string value that specifies if the results of the analysis should be read in after the analysis is complete. Expected values are yes or no.
- **road_data_file:** Hardcoded to BEDPLATE to indicate that the car will not be driven across a road surface. Adams/Car will internally interpret and understand this hardcoded value.
- **generate_road_geometry:** Set to no to indicate that Adams/Car should not generate a geometric representation of the data found in the road_data_file.
- **simulation_type:** Hardcoded to fourpost to indicate that the full-vehicle will be attached to a four-post shaker table. Adams/Car will internally interpret and understand this hardcoded value to produce the correct .adm and .acf files.

Four-Post Vertical Excitation Test

- **Logging the analysis**

- Many users consider it very important to generate a log describing the analysis that is performed. Generation of the log file is important because it provides historical data that can be stored along with the results of the analysis. The stored data can be useful and is sometimes required to allow a user to regenerate the results of a particular analysis. These utility macros are executed from within each of the suspension or full-vehicle analysis macros.
- Utility macros exist that can be used within your custom analysis macro to generate a log file. A utility macro exists for both suspension and full-vehicle analyses. With the analysis completed, the results may be logged to a file. In addition, final diagnostic messages may be displayed to the message window.

Four-Post Vertical Excitation Test

- Finishing up

```
if condition=($log_file)
  acar analysis full_vehicle log &
  assembly=$assembly &
  analysis_name="'output_prefix'_fourpost" &
  analysis_type="Four Post Shaker Test" &
  analysis_log_file="'output_prefix'_fourpost.log" &
  comment=$comment &
  end_time=$end_time &
  number_of_steps=$number_of_steps &
  road_data_file="BEDPLATE" &
  initial_velocity=0.0 &
  velocity_units="m_sec" &
  input_parameters="general input" &
  parameter_values=("$number_of_steps")
end

if condition=("$analysis_mode" != "background")
  acar toolkit message &
  message="Simulation is complete."
end
```

- Finally, it is important to ensure all local variables created in the macro using the \$_self nomenclature are deleted.

```
variable delete variable_name=(eval(db_children($_self,"variable"))
```

Four-Post Vertical Excitation Test

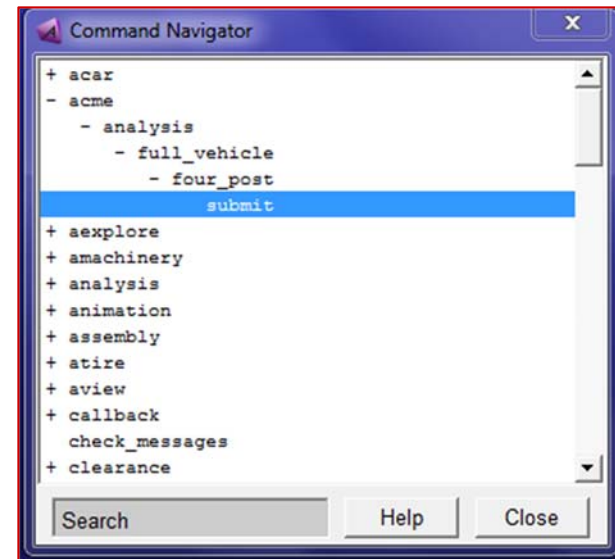
- **Adding the Four-Post Analysis Macro to a Binary**

- You can add the four-post analysis macro to either the private or site binary. The command shown next creates the macro by reading in the command file found at the location you specified. You must place this command either in the acar_build.cmd file or in the macros_ana.cmd file.

```
macro read macro_name=.acme.macros.mac_ana_ful_fou &  
  file_name=(getenv("MDI_ACAR_SITE"))/"/analysis/macros/mac_ana_ful_fou_sub.cmd") &  
  user_entered_command="acme analysis full_vehicle four_post submit" &  
  wrap_in_undo=no &  
  create_panel=no
```

- **Testing the New Analysis Macro**

- The easiest way to test the four-post analysis macro is to access it from the Command Navigator. The command you should use is the user_entered_command specified in the acar_build.cmd file shown next:



Four-Post Vertical Excitation Test

- When you access the four-post macro from the Command Navigator, Adams/Car automatically creates a dialog box based on the parameters in the macro. Users can use this dialog box, shown next, to execute the macro and submit the four-post analysis

The screenshot shows a Windows-style dialog box titled "Acme Analysis Full Vehicle Four Post Submit". It contains the following fields and values:

Field	Value
Assembly	._acme_4PostRig
Output Prefix	example
Comment	An example fourpost testrig analysis
End Time	8
Number Of Steps	800
Peak Displacement	10
Units	mm
Frequency Range	5
Excitation Mode	heave
Analysis Mode	interactive
Load Results	yes
Log File	yes
Error Variable	.ACAR.variables.errorFlag

At the bottom of the dialog are three buttons: "OK", "Apply", and "Cancel".

Four-Post Vertical Excitation Test

- **Customizing the Four-Post Analysis Dialog Box**
 - When you access the four-post macro using the Command Navigator, Adams/Car displays a default dialog box associated with that macro, as shown in the previous slide.
 - The customized dialog box illustrates an example of what you might want to do with the default dialog box generated using the Command Navigator:
 - Change the dialog box title.
 - Remove the Comments and Error Variable text boxes (they will take on their default values as defined in the macro).
 - Modify the Load Results pull-down menu to be radio boxes, and then decrease the area they occupy.
 - Shorten the Brief and Log File pull-down menus to match the Load Results radio boxes.

Four-Post Vertical Excitation Test

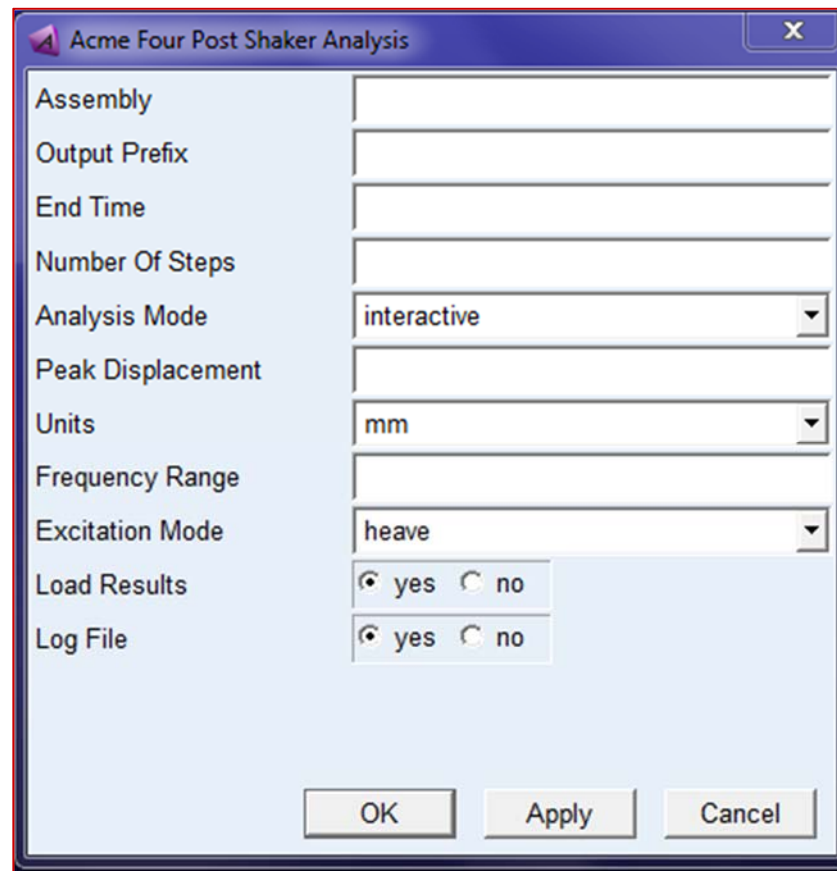
- You can add the four-post custom dialog box to either the private or site binary (or interactively read it in during your Adams/Car session). The command shown next creates the dialog box by reading in the command file found at the location you specify or we have provided an example dialog box in the examples directory (install_dir\acar\examples\fourpost\analysis\dbboxes).
- We have also provided dbboxes_ana.cmd in the example, which contains the following code:

Four-Post Vertical Excitation Test

```
!---- Check if dialog box exists and delete it if it does ----
if condition=(db_exists(".acar.dboxes.dbox_ana_ful_fou_sub"))
    interface dialog_box delete dialog_box_name=.acar.dboxes.dbox_ana_ful_fou_sub
end
!---- Read the Four Post shaker dialog box ----
file command read &
file_name=(getenv("MDI_ACAR_SITE"))//"/analysis/dboxes/dbox_ana_ful_fou_sub.cmd")
!---- Check if menu button exists and delete it if it does ----
if condition=(db_exists(".gui.main.aca_sta_mbar.simulate.full_vehicle_analysis.ACME_FourPost"))
    interface push_button delete &
    push_button_name = .gui.main.aca_sta_mbar.simulate.full_vehicle_analysis.ACME_FourPost
end
!---- Add a menu option ----
interface push_button create &
    push_button_name = .gui.main.aca_sta_mbar.simulate.full_vehicle_analysis.ACME_FourPost &
    enabled = yes &
    help_text = "Simulate a full-vehicle on the ACME FourPost testrig" &
    units = relative &
    horiz_resizing = scale_all &
    vert_resizing = scale_all &
    location = 0.0, 0.0 &
    label = "&ACME Fourpost..." &
    commands = "interface dialog_box display dialog=.acar.dboxes.dbox_ana_ful_fou_sub" &
    default = false
```

Four-Post Vertical Excitation Test

- Customized Dialog Box appears to be one as shown next:



APPENDIX I

SOLUTIONS TO THE WORKSHOP QUESTIONS

Workshop 1 Questions

1. An assembly and a database path for Adams/Car usually ends with what extension ?

The extension for assembly is .asy and a database path has an extension of .cdb

2. If you have an assembly opened in the session and select File - Save, where does the assembly get saved to?

Your default writable database under assemblies.tbl

3. What is the best way to share your entire model with another person?

Publish the model (from the Database Management menu) and share the database

Workshop 2 Questions

1. Template files can be saved in binary or ASCII format – which is the default?

Binary files

2. If the mass property of a part is modified in the subsystem level, where does the change get saved into?

The subsystem file

3. A part in a template has a mass of 30 kg. Then you create a subsystem from that template and change the mass to 40 kg and save the subsystem. Then you modify the template so that the part has a mass of 50 kg and save. When you reload the subsystem, what mass will that part have?

40 kg

Workshop 3 Questions

1. If the wheel front is rotated in towards the vehicle body, Adams sees it as what type of toe/camber?

Positive toe

2. When you change the geometry of a part in a subsystem, will the mass get changed as well?

No it will not be changed automatically

3. List two different methods for adjusting a hardpoint:

Model browser - right click on the hardpoint – Modify

Adjust menu - Hardpoint – Table

Adjust menu - Hardpoint - Modify

Workshop 4 Questions

1. List some of the information that is saved in a plot configuration file:

The mathematical expressions to be displayed for the plot axes. The text/images to be displayed for the header/footer of each page. And the presentation of the plot.

2. What is the name of the testrig that is needed for general suspension analyses?

MDI Suspension Testrig

3. In what interface mode do you create a suspension assembly?

The Standard Interface

Workshop 5 Questions

1. Can the dimensions of imported CAD geometry be parameterized?

No

2. CATIA V5 and Parasolid geometry formats are supported via both import and export: True or False?

True

Workshop 6 Questions

1. Constant Radius Cornering is a form of (circle one) Closed/Open Loop event?

Closed

2. Is a testrig optional while creating a Generic Assembly?

Yes it is optional

Workshop 7 Questions

1. What variables does the Standard Driver Interface (SDI) use to connect with your vehicle model.

Steering, throttle, clutch, gear and brake

2. What file format does the Event Builder save to?

.xml format

3. What is the easiest way to create an event?

Run a standard Full Vehicle Analysis and edit the xml file in the Event Builder

Workshop 8 Questions

1. If a road obstacle is shorter than the tire circumference, which tire contact method should be used?

3D enveloping contact

2. For general handling events, which tire model is the better choice (circle one): PAC2002 / Fiala ?

PAC2002

3. What tire models are the most suitable for durability analyses?

PAC2002, Ftire

Workshop 9 Questions

1. If the road point #3 is defined with 0.9 Friction Left and the Global Parameters section has 0.8 as the Road Friction Left, what friction coefficient does #3 have?

0.9

2. What file format does the Road Builder save to?

The format is .xml

3. What type of road model is the Road Builder used to create?

It creates 3D Spline Roads

Workshop 10 Questions

1. What two file formats can be used to create a Flexible body in Adams?

Modal Neutral File (*.mnf) file or MD DB (*.master) file

2. Can higher order deformations (>10% of CL) be modelled using a single flexible body in Adams Car?

No, you cannot

3. What does it mean if you set the damping ratio value to 1 in "flexible body modify" dialog box?

It means 100% of critical damping

Workshop 11 Questions

1. For a General Actuation Analysis, in what location are the actuation inputs applied?

At the wheel Part

2. What is the "Assembly Class" of a General Actuation Assembly?

It is Actuation

3. Are tires used in a General Actuation Assembly?

No – forces are applied directly to the spindles

Workshop 12 Questions

1. Construction frames specify both location and orientation information.

2. If an assembly match is not found for a mount part, the mount part will be connected to:

Ground

3. Can switch parts be fixed to only one, or several, of the parts on the switch's part list?

Only one part

Workshop 13 Questions

1. If an output communicator does not match with one or more input communicators, what happens?

Just a warning is issued

2. Communicators with the 'inherit' minor role get their role from _____?

The parent subsystem

3. An input communicator, for the left side of your model, having the name 'rack_mount' would have the following name:

cil rack mount

Workshop 14 Questions

1. In what mode (Template Builder or Standard Interface) are REQUESTs created?

In the Template Builder

2. Can you switch REQUESTs ON and OFF in the Standard Interface?

Yes, you can

3. REQUESTs of type 'Define Using Type & Markers' make it easy to measure quantities of type: displacement, velocity, acceleration and force.

4. If you need to create a REQUEST that doesn't fit one of the above categories, what can be done? Create a REQUEST using Function Expressions

Workshop 15 Questions

List two ways to get Info on an existing Communicator in your model:

1. Menus: Build → Communicator → Info
2. Right-click on a Communicator in the Model Browser

2. The Database Navigator can present a visual representation of the connections in your model using the Graphical Topology mode.

3. Can the Database Navigator tell you what PARTs are used by a certain force element in the model?

Yes: use the Associativity mode for this

Workshop 16 Questions

1. Adams Linear output results can be viewed in two main modes, list them:

1. As an animation
2. As tabular data

2. Adams Linear lists the damping ratio as a percentage of what quantity (consult the LINEAR/ command in the Adams Solver guide for clues..):

A percentage of critical damping for that mode

3. Can your Adams model be linearized about different operating points? Yes / No

Workshop 17 Questions

1. What is the upper limit of number of axles that the tilt table testrig can support?

The upper limit is six

2. List two required inputs for the Tilt Table Analysis?

The tilt rate and maximum angle, the force sensor threshold, and the move right side

Workshop 18 Questions

1. Adams Car recognizes these two types of model element as Factors in Adams Insight:

Hardpoints & parameter variables

2. When reviewing response surface fit indicators, which color indicates a good fit?

Green

3. Can Objectives (Responses) be an array of elements?

Yes

Workshop 19 Questions

1. What is the basic difference between half and full vehicle SVC?

In addition to suspension characteristics, full vehicle SVC computes general characteristics such as ground reactions and mass properties

2. How many kinematic and compliance sub-events does SPMM have?

Five

3. What type of analysis is SPMM?

Quasi-static: all loads and displacements are applied quite slowly. This is done to look at force-displacement relationships in detail.

